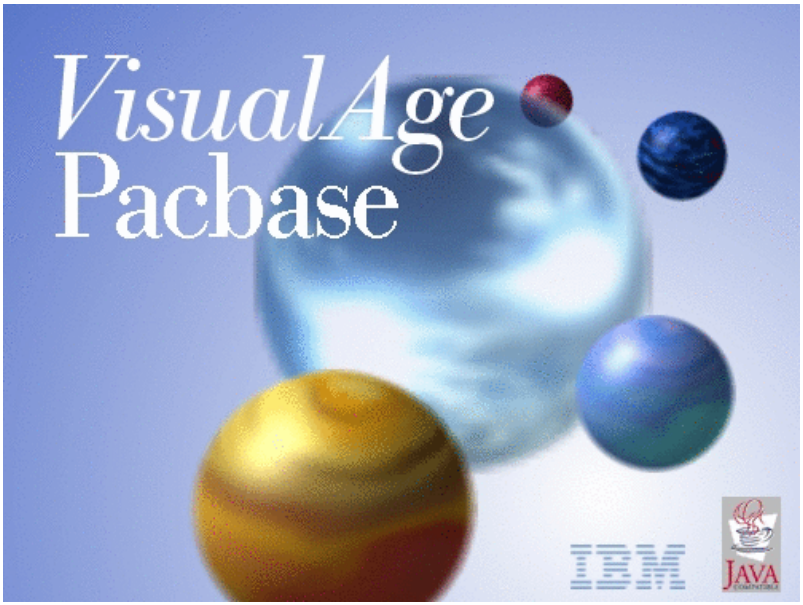
VisualAge Pacbase

PACBASE ACCESS FACILITY

Version 3.5



VisualAge Pacbase

PACBASE ACCESS FACILITY

Version 3.5

Note

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section «Notices», à la page v.

Vous pouvez consulter ou télécharger la documentation de VisualAge Pacbase, régulièrement mise à jour, à partir de :

http://www.ibm.com/support/docview.wss?rs=37&context=SSEP67&uid=swg27005478

La section Catalogue dans la page d'accueil de la Documentation vous permet d'identifier la dernière édition disponible du présent document.

Première édition (Décembre 2006)

La présente édition s'applique à :
• VisualAge Pacbase Version 3.5

Vous pouvez nous adresser tout commentaire sur ce document (en indiquant sa référence) via le site Web de notre Support Technique à l'adresse suivante : http://www.ibm.com/software/awdtools/vapacbase/support.html ou en nous adressant un courrier à :

IBM Paris Laboratory
1, place Jean–Baptiste Clément
93881 Noisy-le-Grand, France.

IBM pourra disposer comme elle l'entendra des informations contenues dans vos commentaires, sans aucune obligation de sa part.

Table des matières

Notices

Ce document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM. Cela ne signifie pas qu'IBM ait l'intention de les annoncer dans tous les pays où la compagnie est présente. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante : IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk NY 10504–1785, U.S.A.

Les détenteurs de licences du présent produit souhaitant obtenir des informations sur celui-ci à des fins : (i) d'échange d'informations entre des programmes développés indépendamment et d'autres programmes (y compris celui-ci) et (ii) d'utilisation mutuelle des informations ainsi échangées doivent s'adresser à : IBM Paris Laboratory, SMC Department, 1 place J.B.Clément, 93881 Noisy-Le-Grand Cedex, France. De telles informations peuvent être mises à la disposition du Client et seront soumises aux termes et conditions appropriés, y compris dans certains cas au paiement d'une redevance.

IBM peut modifier ce document, le produit qu'il décrit ou les deux.

Marques

IBM est une marque d'International Business Machines Corporation, Inc. AIX, AS/400, CICS, CICS/MVS, CICS/VSE, COBOL/2, DB2, IMS, MQSeries, OS/2, VisualAge Pacbase, RACF, RS/6000, SQL/DS et VisualAge sont des marques d'International Business Machines Corporation, Inc. dans certains pays.

Java et toutes les marques et logos incluant Java sont des marques de Sun Microsystems, Inc. dans certains pays.

Microsoft, Windows, Windows NT et le logo Windows sont des marques de Microsoft Corporation dans certains pays.

UNIX est une marque enregistrée aux Etats-Unis et/ou dans d'autres pays et utilisée avec l'autorisation exclusive de la société X/Open Company Limited.

D'autres sociétés peuvent être propriétaires des autres marques, noms de produits ou logos qui pourraient apparaître dans ce document.

Chapitre 1. Présentation du module PAF

Introduction

Le module PAF (Pacbase Access Facility) permet d'extraire par programme les informations contenues dans les Bases de développement et la Base administration à l'aide de requêtes SQL. Ces données extraites peuvent ensuite être utilisées pour mettre à jour une base VisualAge Pacbase.

VISION RELATIONNELLE DU METAMODELE VA PAC

PAF fournit une description relationnelle du métamodèle VA Pac, préalable indispensable à l'expression d'une requête SQL. Un modèle relationnel organise les données dans des tables structurées en colonnes. Les relations entre objets sont des correspondances entre des colonnes de tables.

Dans le modèle tabulaire PAF,

- Chaque entité est décrite par un ensemble de tables,
- Les informations relatives à la définition et à la description de chaque entité sont décrites par des colonnes,
- Les chaînages peuvent être décrits de deux manières :
  - Par des relations directes entre des colonnes de tables,
  - Par des tables dites ″virtuelles″ dont les colonnes sont, d'une part les colonnes identifiant les deux tables à mettre en relation, d'autre part les colonnes représentatives du chaînage.

  >>>> Voir le sous-chapitre suivant pour une description complète du modèle tabulaire PAF.

PRESENTATION DU LANGAGE SQL

Le langage SQL (pour Structured Query Language) est actuellement le langage standard d'interrogation des bases de données relationnelles. Il permet de formuler des requêtes dans une syntaxe proche de la langue anglaise, de récupérer l'information ainsi définie et de mettre à jour une base.

Une requête SQL est définie par une clause SELECT (voir le sous-chapitre ″Syntaxe du Langage PAF-SQL″). Les tables d'une base de données relationnelle sont constituées de rangs (assimilables à des enregistrements pour les fichiers).

Une requête SQL définit un sous-ensemble parmi les informations d'une base, sans préciser a priori quand ou comment ces informations sont obtenues et traitées. A ce titre, SQL peut être considéré comme un langage "ensembliste".

Afin d'intégrer SQL dans les langages procéduraux utilisés en informatique de gestion, ce langage permet d'associer la notion de curseur à une requête. Le curseur permet de préciser les accès à l'information définie dans la requête associée lors de la déclaration du curseur.

SQL fournit donc les ordres de manipulation de curseur permettant de lire dans la base les informations définies par la requête (OPEN), de retourner un par un les rangs dans une zone de travail du programme utilisant SQL (FETCH) et enfin de terminer l'utilisation du curseur (CLOSE).

Ainsi, une fois le curseur déclaré, les ordres de manipulation de curseur permettent de voir un curseur comme un fichier, l'ordre FETCH étant l'équivalent du READ en COBOL.

REQUETES SQL STANDARD

Avec le module PAF, les bases de développement et la base administration deviennent accessibles par le langage de requêtes standard dans le monde des bases de données relationnelles.

A partir du modèle tabulaire, décrit dans le sous-chapitre suivant, les accès à ces bases s'effectuent par la déclaration et l'utilisation de curseurs SQL. A chaque curseur est associé un contexte VA Pac (Bibliothèque, session). Il est possible d'exploiter plusieurs curseurs pour accéder à des informations mettant en jeu plusieurs chaînages VA Pac.

Les requêtes peuvent être décrites dans tous les programmes utilisateur, batch ou TP, écrits directement en COBOL ou générés par VA Pac. Dans tous les cas, les requêtes sont traitées par le translateur PAF, en amont du compilateur COBOL, qui traduit les requêtes en ordres d'appel du sous-programme d'extraction PAF.

UN SOUS-PROGRAMME D'EXTRACTION

Un sous-programme PAF gère les accès aux bases de développement et d'administration. Il récupère les paramètres internes construits par le translateur lui permettant de réaliser l'extraction demandée.

Les informations extraites sont transmises au programme utilisateur dans la zone de communication générée par le translateur (zone de communication COBOL en Working Storage Section).

Le modèle tabulaire PAF

INTRODUCTION

Le modèle tabulaire PAF est un modèle relationnel qui organise les données dans des tables structurées en colonnes. Ces colonnes sont définies par des Rubriques VA Pac et les tables sont associées à des Segments ou à des Méta-Entités extension.

Toutes les entités nécessaires à l'élaboration du modèle tabulaire sont fournies sous forme de mouvements batch (procédure UPDT) permettant la mise à jour d'une Bibliothèque VA Pac :

- les définitions des Rubriques, représentant les colonnes des tables PAF,
- les définitions des Structures de Données, regroupant les tables par famille d'entités VA Pac,
- les définitions des Segments et Méta extensions supportant les tables PAF,
- la description (liste des Rubriques) de ces Segments et de ces Méta Entités.

Toutes ces entités comportent le mot-clé 'PAF'.

CODIFICATION ET DESCRIPTION DES TABLES

Les tables ont en règle générale une codification mnémonique sur 6 à 10 caractères, les 3 premiers caractères identifiant l'entité concernée.

C'est ce code qui doit être spécifié dans la déclaration du curseur : SELECT * FROM <code-table>.

Note : La Documentation des Tables PAF est disponible à l'adresse suivante :

www-1.ibm.com/support/docview.wss?rs=37&context=SSEP67 &uid=swg27005478.

Cliquez sur le choix "Module PAF". Des liens vous permettent alors de vous débrancher sur les listes des tables PAF de toutes les entités, regroupées par unité fonctionnelle, c'est-à-dire :

- les entités du métamodèle classique VA Pac,
- les entités du modèle eBusiness,
- les entités gérées dans l'espace administration et les entités de gestion Administrateur liées à une base de développement,
- les entités Méthodes.

Vous trouverez également à cette adresse un index cumulatif des tables de toutes les entités, trié par code table. A partir de cet index, vous pourrez vous débrancher directement sur la description de la table recherchée.

TABLES VIRTUELLES

Les tables dites ″virtuelles″ représentent une partie des chaînages sur les entités. Elles sont codifiées en intercalant un 'X' entre les 2 entités chaînées pour VisualAge Pacbase.

Ces tables sont introduites dès lors qu'un chaînage ne peut être défini trivialement par une condition élémentaire SQL

EXEMPLE : L'appel d'une Rubrique dans une ligne -P de Programme est un chaînage créé dès lors que le code de la Rubrique apparaît dans la zone opérande ou dans la zone condition. Ces deux zones banalisées pouvant contenir, outre le code Rubrique, un autre opérande (par exemple) on ne peut en SQL décrire le chaînage à partir des seules colonnes de la table représentant les lignes -P des Programmes, PGMSPE.

Il est donc nécessaire de créer une nouvelle table pour implémenter ce chaînage : RUBXPGMSPE. Cette table est composée des colonnes identifiantes des deux entités chaînées (code Rubrique, code Programme, code fonction, code sous-fonction et numéro de ligne) ainsi que des autres colonnes de la table PGMSPE.

Une table virtuelle contient au minimum les colonnes identifiantes des deux entités.

Entités du métamodèle classique VA Pac

Dans le métamodèle classique VA Pac, les 3 premiers caractères du code table identifiant l'entité sont :

```
Base                        BAS
Bibliothèque                BIB
Bloc Base de Données        DBD
Dialogue                    DIA
Ecran                       ECR
Etat                        ETA
Format Guide                FOG
Merise :
          C.I.F.            CIF
          Objet             OBJ
          Relation          REL
Programme                   PGM
Rapport                     RAP
Rubrique                    RUB
Rubrique non définie        NDE
Segment                     SEG
```

```
Structure de Données        SDO
Texte                       TXT
Meta-entité cliente         MEC
Meta-entité extension       MEE
Relation Utilis. cliente    RLC
Relation Utilis. extension  RLE
Entité utilis. cliente      $TT
Entité utilis. extension    YTT
```

Les 3 caractères suivants précisent l'élément du dossier de l'entité :

```
Définition                        DEF
Commentaire                       DOC
Utilisation dans Desc. de Rapport RAP
Textes ventilés                   TXT
Description                       DSC
Description Entité Util.          Dxx
Description en-tête (Méta Entité) DSE
Description hiérarchique          DSH
Description Codasyl               DSC
Description relationnelle         DSR
Aide en ligne                     ERR
Génération                        GEN
Paramètres                        OPT
Appel par relation util cliente   APP
Appel par relation util extension APE
Appel de Programmes               PGM
Appel de Rubriques                RUB
Appel de Segments                 SEG
Appel de S. D.                    SDO
Appel d'Objet Merise              OBJ
Zones de Working                  WSS
Modif. Début Programme            DEB
Traitements spécifiques           SPE
Traitements spécifiques 8         SP8
Appel de macros                   MSP
Cobol (lignes 9)                  COB
Lignes source                     SRC   (REVERSE)
Complément                        CMP   (Dialogue)
Libellés s-sch. s-sys.            TAB   (Segment)
Libellés                          LIB   (Etat)
Structures                        STR   ( "  )
Catégories                        CAT   ( "  )
```

Exemples de codes tables pour l'entité Rubrique

```
RUBDEF      DEFINITION DES RUBRIQUES
RUBDSC      DESCRIPTION DES RUBRIQUES
RUBXTXT     UTI. DANS DSC DE TEXTES
            (chaînage des Rubriques utilisées dans les
             Descriptions des Textes).
```

Cas particuliers

```
BLOCAGE      Liste des entités bloquées
SESHIS       Liste des sessions historisées
HORODAT      Liste des horodatages
MCLDEF       Liste des mots-clés du thésaurus
MOTCLE       Recherche par mots-clés
```

CODIFICATION DES COLONNES DES TABLES

Les colonnes des tables sont des Rubriques, donc codifiées sur 6 caractères. La règle adoptée est la suivante :

- le 1er caractère représente la nature de la donnée : code, libellé, type, option, ...
- les 3 caractères suivants précisent l'objet ou le centre d'intérêt de la donnée : entité, fonction, ligne, ...
- enfin, les 2 derniers caractères permettent d'ajouter un qualificatif : mère (pour un code Rubrique), précision d'un type ou d'une option (entrée, interne ou sortie pour un format), ...

Les 4 premiers caractères ont donc une codification précise, dont le détail est donné ci-dessous.

Les 2 derniers caractères n'ont pas de signification intrinsèque, ils prennent leur sens en fonction des objets qu'ils décrivent. Par exemple, le suffixe 'MO' pourra signifier 'MOYEN' (NRELMO : cardinalité moyenne d'une relation) ou 'MODIFICATION' (VMVTMO : valeur du mouvement en modification).

DEFINITION DES DIFFERENTES NATURES

C : code ou numéro (au sens d'un identifiant)

L : libellé, nom en clair

E : nom externe

T : type

O : option

N : nombre, niveau

F : format, longueur

A : adresse, position

D : documentation, description ou commentaire

V : valeur, contenu

DEFINITION DES OBJETS OU CENTRES D'INTERET

LIG : ligne

ENR : enregistrement

MVT : mouvement

STE : structure d'enregistrement

SSS : sous-schéma ou sous-système

LIB : libellé

STR : structure

CAT : catégorie

FON : fonction

SFO : sous-fonction

CND : condition

ATR : attribut de Rubrique

LAN : langage (génération)

CAV : cartes de contrôle avant (génération)

CAP : cartes de contrôle après (génération)

SEC : section

PAR : paragraphe

DIV : division

SET : set

CHA : niveau 1 (anciennement ″chapitre″)

SCH : niveau 2 (anciennement ″sous-chapitre″)

ENT : entité

SEN : sous-entité

SES : session

UTI : utilisateur

BLO : blocage d'entité

DSC : description

SEL : sélection

VER : version

La codification des entités VA Pac est la même que celle des codes des tables (RUB, OBJ, REL, ...) à l'exception des Relations Utilisateur Extensions qui sont codifiées ″RLE″ dans les TABLES et ″RLX″ dans les COLONNES.

REMARQUE : Ces règles ne s'appliquent pas aux entités gérées par la Station de Travail. Les colonnes sont les rubriques composant les Méta Entités dédiées à la Station de Travail VA Pac. Il en est de même pour les entités définies par les Méta Extension (Administration et eBusiness).

Entités Méthodes

Les tables associées aux entités gérées par la Station de Travail VA Pac sont un cas particulier des tables d'Entité.

Une entité Station est décrite par une ou plusieurs Meta-Entités dans la base de spécifications.

Les trois premiers caractères du code de la table identifient l'entité concernée. Dans ce cas, c'est donc le type d'appel de chacune de ces Méta-Entités qui va servir d'identifiant.

Les trois caractères suivants précisent l'élément du dossier de l'entité: DEF, DOC, TXT, APP, APE et Dxx.

Les tables correspondant aux Descriptions des entités gérées par la Station sont identifiées par Dxx où xx est l'identifiant de la Description de la Meta-Entité correspondante.

EXEMPLE : Le code de la table correspondant à la Définition d'un MCD est $1SDEF.

Pour connaitre les types d'appel des Méta-Entité dédiées à la station de travail et les identifiants des descriptions spécifiques à chacune des méthodes, utilisez la commande PCM saisie dans la zone COM de l'écran ″COMMANDES D'EDITION ET DE GENERATION″ (CH: GP). Le code de la méthode sur un caractère est saisi dans la zone ENTITE.

Liste des valeurs possibles pour les Méthodes :

M pour Merise

D pour YSM

A pour SSADM

O pour OMT

F pour IFW

Entités d'Administration et de Gestion

Entités Administration

Pour ces entités gérées dans l'espace administration, (communes à plusieurs bases), les 3 premiers caractères du code table identifiant l'entité sont les suivants :

```
Base                        YAB
Clé d'accès                 YAK
Ligne de commande           YAF
Pactransfer                 YAR
Paramètre                   YAT
Procédure                   YAP
Profil                      YAV
Sécurité                    YAS
Utilisateur                 YAU
Endevor                     YEN
```

Entités de Gestion

Pour ces entités, gérées dans une base de développement, les 3 premiers caractères du code table identifiant l'entité sont les suivants :

```
Plan type d'Extraction      Y7E
Définition règle qualité    Y5Q
Commande edition-génération YG1
Environnement               YC1
```

Entités eBusiness

Dans le modèle eBusiness, les 3 premiers caractères du code table identifiant l'entité sont les suivants :

Application	YCS
Composant Elémentaire	YCE
Dossier	YDO
Message	YMS
Moniteur Communication	YMC
Opération	YOP
Part	YPT
Service	YSV
Liaison SOAP	YSB
Serveur d'Initialisation	YSI
Vue Logique	YVL

Chapitre 2. Fonctionnement et utilisation dans les Programmes

Fonctionnement général

LE PROGRAMME UTILISATEUR

La mise en oeuvre de PAF est possible à travers un programme batch ou conversationnel, écrit directement en COBOL ou généré par VA Pac. Dans tous les cas, c'est le sous-programme d'extraction PAF qui gère les accès à la base de spécifications.

Dans son programme, l'utilisateur doit d'abord déclarer (DECLARE CURSOR) le ou les curseurs lui permettant l'accès aux tables souhaitées. Ensuite, pour chaque curseur déclaré, la séquence d'ordres est la suivante :

CONNECT

Connexion du curseur à un contexte VisualAge Pacbase (Utilisateur, Bibliothèque et session),

OPEN

Ouverture du curseur, c'est-à-dire extraction préliminaire des informations de la base,

FETCH

Restitution, un par un, des enregistrements extraits,

CLOSE

Fermeture du curseur.

L'ordre SET permet la modification dynamique des paramètres de fonctionnement du translateur.

Enfin, les ordres INIT et QUIT permettent d'effectuer les actions d'initialisation et de terminaison techniques en fonction du mode d'extraction et du type de matériel (par exemple ouverture et fermeture de fichiers en batch). Ils ne sont pas obligatoires en TP.

LE TRANSLATEUR PAF

Les appels à PAF ne sont donc pas écrits en pur COBOL, des ordres spécifiques en langage SQL sont insérés dans le programme utilisateur. Le rôle du translateur est donc de transformer les phrases SQL-PAF en COBOL avant la compilation. Tout comme le préprocesseur CICS par exemple, le translateur PAF place en commentaire les phrases SQL-PAF et les traduit en COBOL à la suite de ces commentaires.

Le translateur traduit les ordres SQL-PAF en déclarations (pour l'ordre DECLARE) et en appels au sous-programme d'extraction pour les autres ordres (sauf pour un ordre très spécifique et non SQL : l'ordre SET).

Ces zones permettent également à l'extracteur de retourner l'information demandée dans la requête SQL-PAF. Une description plus détaillée des zones déclarées et des différents ordres SQL-PAF est donnée plus loin.

Le translateur édite un compte rendu des erreurs décelées.

Le translateur est paramétré par une carte commentaire insérée en ligne 2 (après la ligne IDENTIFICATION DIVISION). Cette ligne, décrite ci-dessous, est générée automatiquement par le générateur Va Pac à la rencontre d'un opérateur EXP dans une ligne de code source (-P) du programme (voir le sous-chapitre ″Mise en oeuvre sous VA Pac″).

La ligne de paramètres a le format suivant (à partir de la colonne 8, la colonne 7 contenant l'astérisque des commentaires COBOL) :

POS	LONG	CONTENU
1	6	Numéro de ligne COBOL
7	1	Commentaire
8	5	Mode d'exécution (BATCH ou TP)
14	4	Libellé fixe
19	3	Code bibliothèque
23	5	Numéro de session - état de version
29	2	Variante(s) de génération (COBOL et MAP)
32	3	Libellé fixe
36	1	Code langue de la base (F ou A)
38	3	Squelette programmes Langage batch, Dialogue ou programmes Gén. COBOL
42	1	Langage du squelette (F ou A)
44	6	Délimiteur (SINGLE ou DOUBLE)

Le mode d'exécution permet de distinguer le batch du TP. Le mode pris par VA Pac est simplement fonction du générateur mis en oeuvre. Ce mode permet au translateur de déclarer ou non des zones de travail spécifiques au TP et de générer les appels au bon sous-programme d'extraction.

La variante de génération est prise par VisualAge Pacbase au niveau de la fiche Programme. Elle permet d'adapter la syntaxe générée en fonction du compilateur.

Le délimiteur de chaîne, pris par VA Pac dans la fiche Bibliothèque, permet au translateur de connaître le bon délimiteur de chaîne, à la fois pour la génération et pour l'analyse du source PAF.

Les paramètres Bibliothèque et Session sont les paramètres permettant au translateur PAF de se connecter à la base VisualAge Pacbase lorsqu'il traite un curseur concernant une Entité Utilisateur.

Les colonnes des tables associées aux Entités Utilisateur dépendent en effet de la description de la Meta-Entité correspondante. Le translateur doit donc lire cette ME pour valider la requête. Les valeurs des paramètres Bibliothèque et Session sont celles de la génération du programme par VisualAge Pacbase.

Notons que le translateur est un programme bilingue. Le premier code langue spécifie la langue des messages d'erreur du translateur et le second la langue de représentation du Dictionnaire (langue de codage des tables et des colonnes). Ces deux codes langues permettent à des sites de générer des programmes pour des sites travaillant dans une autre langue.

Les valeurs prises par VisualAge Pacbase pour ces deux codes langues sont respectivement ceux des fichiers AE et SG (libellés d'erreur et squelette de génération).

La ligne 2 générée par le générateur VA Pac est lue par le translateur PAF mais n'est pas reproduite dans le source traduit.

L'ordre SET permet de modifier facilement certains de ces paramètres à n'importe quel moment de la traduction. Par exemple, il peut être utile de modifier plusieurs fois la Bibliothèque. Toutefois, dans le cas d'un programme TP contenant uniquement des ordres INSERT et FETCHER (sans DECLARE CURSOR), il faut impérativement que l'ordre SET précède ces 2 ordres pour des raisons d'initialisation de zones (par exemple, écrire l'ordre en Working Storage Section).

L'EXTRACTEUR PAF

Le sous-programme extracteur gère les accès aux bases de développement et d'administration.

Il récupère les paramètres internes construits par le translateur (zone de spécifications où la requête SQL est traduite) et réalise l'extraction demandée de la façon suivante :

- A la rencontre d'un ordre CONNECT, il établit la connexion, pour le curseur précisé, de l'utilisateur au réseau VA Pac (vérification des droits d'accès, Bibliothèque et session).
- A la rencontre d'un ordre OPEN, il accède à la base de Développement ou d'administration et stocke les enregistrements extraits dans le fichier intermédiaire de travail. Le nombre d'enregistrements lus en avance et stockés dans le fichier de travail est paramétrable pour chaque curseur (paramètre SIZE dans l'ordre CONNECT).
- A la rencontre d'un ordre FETCH, l'extracteur restitue un par un les enregistrements stockés dans le fichier intermédiaire (fetching d'un enregistrement). Les enregistrements sont transmis au programme utilisateur dans la zone de communication générée par le translateur.

  Le sous-chapitre ″Optimisation des accès à la base″ explique plus en détail le mécanisme utilisé par l'extracteur pour gérer les ordres OPEN et FETCH.
- L'ordre CLOSE provoque la fermeture du curseur précisé.

LE FICHIER INTERMEDIAIRE DE TRAVAIL

Ce fichier sert à stocker les enregistrements extraits des bases de développement ou d'administration suite à la rencontre d'un ordre OPEN.

Les enregistrements stockés sont ensuite restitués, un par un, à la rencontre de chaque ordre FETCH.

Le nombre maximum d'enregistrements lus en avance pour un curseur est paramétrable (paramètre SIZE de l'ordre CONNECT, se reporter aux sous-chapitres ″Syntaxe du Langage PAF-SQL″ et ″Optimisation des Accès à la Base″).

Outre les données lues en avance, le fichier intermédiaire permet de stocker des paramètres techniques pour la gestion des curseurs par l'extracteur.

Le fichier intermédiaire de travail est un fichier séquentiel indexé au format variable. Les enregistrements qui y sont stockés sont liés à un utilisateur et à un curseur SQL.

La structure de la clé est légèrement différente en batch et en TP :

Traitements batch

Le fichier intermédiaire de stockage est défini pour un job batch.

Clé d'accès

. Code du curseur,

. Code structure,

. Numéro d'enregistrement.

Traitements conversationnels

Un fichier de travail unique est défini pour toutes les conversations, tous utilisateurs confondus.

Clé d'accès

. Identifiant de la conversation,

. Code du curseur,

. Code structure,

. Numéro d'enregistrement.

Cas particulier de CICS

L'environnement conversationnel CICS permettant la coexistence de plusieurs bases VisualAge Pacbase, un fichier de travail est défini pour chacune des bases sous CICS.

Syntaxe du langage PAF-SQL

GENERALITES SUR LE LANGAGE

Les accès aux bases de développement et d'administration par le module PAF se font par la déclaration et l'utilisation de curseurs SQL.

Pour chaque table à laquelle il veut accéder, l'utilisateur doit d'abord déclarer un curseur en WORKING STORAGE SECTION (DECLARE CURSOR). Ensuite, en PROCEDURE DIVISION, la séquence des ordres associée à un curseur donné est la suivante : CONNECT, OPEN, FETCH et CLOSE. OPEN, FETCH et CLOSE sont des ordres standard du langage SQL, tandis que CONNECT est spécifique à PAF.

Ces quatre ordres sont désignés dans la suite comme les ordres de manipulation de curseurs dans la syntaxe PAF-SQL.

Il est possible d'utiliser jusqu'à 100 curseurs dans un programme applicatif.

Enfin, les ordres INIT (initialisations) et QUIT (terminaisons), indépendants des curseurs, doivent être exécutés respectivement avant et après tous les autres ordres de manipulation de curseur (obligatoires en BATCH).

Toute phrase PAF-SQL doit commencer en colonne 12 du COBOL, débuter par EXEC PAF et se terminer par END-EXEC.

DECLARATION D'UN CURSEUR

Ecrite en WORKING STORAGE SECTION, la clause DECLARE CURSOR permet de déclarer un curseur SQL.

Dans cette déclaration, les mots-clés retenus sont SELECT, FROM, WHERE, AND et OR.

Sa syntaxe est la suivante (les parties entre parenthèses sont facultatives) :

EXEC PAF DECLARE <code-curseur> CURSOR FOR

SELECT * FROM <code-table> (WHERE <liste de conditions>)

END-EXEC

avec :

- <code-curseur> est le code du curseur sur 4 caractères,
- SELECT ne permet pas la sélection de colonnes, autrement dit l'accès à une table ramène l'intégralité de ses colonnes dans le programme utilisateur. La seule syntaxe autorisée est donc "SELECT *".
- FROM n'autorise pas la demande d'une jointure de tables et n'est donc suivi que par un seul code table.
- <code-table> est le code de la table PAF.
- WHERE n'autorise pas les SUBSELECT.
- <liste de conditions> est une suite de conditions, chacune portant sur une colonne de la table, reliées entre elles par des parenthèses et les connecteurs logiques AND et OR. Le nombre de conditions élémentaires est limité à 50. Chaque condition s'écrit sous la forme COLONNE OPERATEUR OPERANDE, où :
  - COLONNE est le code d'une colonne de la table,
  - OPERATEUR est un des opérateurs suivants :

- = égal,
- > supérieur,
- >= supérieur ou égal,
- < inférieur,
- <= inférieur ou égal,
- <> différent.

– OPERANDE est soit :
  - une autre colonne de la table,
  - une constante COBOL,
  - une constante numérique ou alphanumérique,
  - une variable COBOL du programme.

  WHERE COEU > ..... AND COEU < ..... .

Les contraintes sur la clause SELECT ne sont pas restrictives : d'une part, la possibilité d'ouvrir plusieurs curseurs pallie l'inconvénient des accès monotables ; d'autre part, il est souvent délicat d'exprimer une requête très complexe en SQL. Enfin, l'objectif de PAF n'est pas de fournir une interface SQL exhaustive mais de permettre d'accéder à toute l'information contenue dans les Bases de Développement et d'administration. C'est au niveau du programme utilisateur que sont structurés les accès à l'information.

Pour de plus amples informations, voir le sous-chapitre "Imbrication des Curseurs PAF" dans ce chapitre.

MANIPULATION D'UN CURSEUR

Les ordres de manipulations de curseurs s'écrivent en PROCEDURE DIVISION.

CONNECT

Premier dans la séquence des ordres de manipulation d'un curseur, l'ordre CONNECT permet d'établir la connexion à une base de développement ou à la base administration VA Pac. A tout moment, il est possible de connecter ou de re-connecter un curseur, pour modifier le contexte de connexion par exemple.

Syntaxe :

```
EXEC PAF  CONNECT <code-curseur> TO
   USER = <code-utilisateur>
   Le code Utilisateur doit avoir les droits administrateur
   pour une connexion à la base administration.
   PASS = <mot-de-passe>
```

```
    LIB = <code-bibliothèque>
        = <***> dans l'espace administration
    SESSION = <session-et-version>
        = <0001Z> dans l'espace administration
    NET = <option-sous-réseau>
    SIZE = <nombre>
    IDENT = <identifiant-TP>
    BASE = <code-base>
        = pas de paramètre Base dans l'espace administration
END-EXEC
```

Les paramètres désignés à la droite du symbole = et décrits ci-dessous doivent être soit des littéraux, soit des variables du langage COBOL.

USER : code utilisateur VisualAge Pacbase,

PASS : mot de passe VisualAge Pacbase,

LIB : code Bibliothèque VisualAge Pacbase,

SESSION : numéro de session VA Pac et version dans la session,

NET : sous-réseau sélectionné : I, C, A, U, <, > ou Z,

SIZE : nombre maximum d'enregistrements stockés dans le fichier intermédiaire de travail,

IDENT : identifiant banalisé de conversation en TP,

BASE : code transaction VA Pac sur 4 caractères.

Tous les paramètres du CONNECT sont obligatoires (à l'exception du paramètre Base dans l'espace administration) lors de la première connexion de chaque curseur. En effet, l'ordre CONNECT permet d'affecter des valeurs aux paramètres associés au curseur pendant l'extraction. Cependant, il est possible de modifier certains paramètres lors d'une connexion ultérieure (par exemple pour examiner la même requête en changeant le contexte VisualAge Pacbase d'exploration).

Il est toujours possible de reconnecter un curseur (par exemple pour explorer un autre sous réseau avec la même requête). Afin de permettre une reconnexion souple, il n'est pas obligatoire de préciser l'ensemble des paramètres dans la syntaxe, ni même de respecter l'ordre standard présenté ci-dessus. Ainsi, la syntaxe

EXEC PAF CONNECT <code-curseur> TO

NET = <option-sous-réseau>

LIB = <code-bibliothèque>

END-EXEC

est parfaitement correcte. Il faut cependant s'assurer que l'ensemble des paramètres ont une valeur correcte lors de la première connexion de chaque curseur.

REMARQUES : PARAMETRES IDENT ET BASE

Les deux paramètres IDENT et BASE ne sont utiles qu'en TP. Par conséquent, le translateur PAF les ignore lors de la précompilation d'un programme utilisateur de type BATCH.

Le paramètre IDENT est une zone banalisée de 25 caractères permettant d'identifier la conversation TP utilisant PAF. Elle fait partie de la clé d'accès au fichier intermédiaire. La valeur prise par ce paramètre dépend naturellement du moniteur TP. Par exemple, sous CICS, il est recommandé d'alimenter ce paramètre à l'aide de la variable DFHTERMID fournie en standard par CICS.

Un ensemble de recommandations pour alimenter de façon standard la variable IDENT en fonction du système d'exploitation est proposé dans le sous-chapitre ″Recommandations pour le paramètre IDENT″.

Le paramètre BASE contient le code transaction d'accès à la base VA Pac sur quatre caractères. Il permet l'accès en TP aux fichiers d'une base de développement VA Pac ainsi qu'aux fichiers de travail PAF.

Ce paramètre n'est pas requis pour un accès à la base administration.

Note : Ce paramètre n'est utilisé que pour le matériel IBM sous CICS.

REMARQUES : PARAMETRE SIZE

1. Dépassement du SIZE

Le nombre maximum d'enregistrements du fichier intermédiaire peut être dépassé dans le cas des tables :

- descriptions de Segments (SEGRUB),

- Commentaires des entités (entDOC) ou des descriptions d'entités (entdscDOC), lorsque figurent des appels de Formats Guides.

Pour ces tables, l'extraction se poursuit jusqu'à la fin de la description du Segment ou du Format Guide en cours. Ensuite, elle est stoppée si le nombre maximum (SIZE) est atteint ou dépassé.

2. Contrôles sur le paramètre SIZE

Le paramètre SIZE doit être alimenté par une quantité strictement positive. Dans le cas contraire, il est repositionné à la valeur arbitraire 1 par l'extracteur PAF.

OPEN

A la rencontre d'un ordre OPEN, l'extracteur accède à la base VA Pac et écrit, dans le fichier intermédiaire de stockage, les enregistrements sélectionnés en fonction de la déclaration du curseur.

Un curseur peut être ouvert s'il a été connecté auparavant et s'il n'a pas été déjà ouvert.

Syntaxe :

EXEC PAF OPEN <code-curseur> END-EXEC

FETCH

A la rencontre d'un ordre FETCH, l'extracteur restitue au programme utilisateur un enregistrement du fichier intermédiaire.

Le curseur doit être ouvert ; pour ce curseur, l'utilisateur écrit autant de FETCH que nécessaire.

Syntaxe :

EXEC PAF FETCH <code-curseur> END-EXEC

Remarque :

La syntaxe standard de l'ordre FETCH dans le langage SQL prévoit le mot-clé INTO, permettant d'associer à chaque colonne sélectionnée dans le SELECT du DECLARE CURSOR une zone COBOL.

L'extracteur PAF-SQL restituant l'ensemble des colonnes dans une zone de communication, générée en WORKING STORAGE SECTION à la rencontre de

l'ordre DECLARE, le mot-clé INTO, suivi de la liste des zones COBOL cibles, n'est pas conservé dans la syntaxe de l'ordre FETCH.

CLOSE

Il n'est possible de fermer qu'un curseur ouvert. L'ordre CLOSE suit donc un ordre OPEN ou un FETCH.

Syntaxe :

EXEC PAF CLOSE <code-curseur> END-EXEC

INSERT

L'ordre INSERT permet d'insérer des données dans le fichier intermédiaire de travail (par exemple une extraction PAF). Ces données devront être préalablement stockées dans la zone pref-PUFENR de pref-PAFCOM - ″pref″ étant un préfixe de 1 à 4 caractères -

Pour la mise à jour, il faut que toutes les zones de l'enregistrement à modifier soient renseignées : en effet, dans le cas contraire, elles seraient remises à blanc (contrairement aux procédures batch UPDT et UPDP).

Les différentes valeurs possibles du code mouvement sont : 'C' pour une création, 'M' pour une modification , 'A' pour une annulation et 'B' pour une annulation multiple. Cette valeur est stockée dans la zone pref-PUFMVT de pref-PAFCOM.

Remarque : Lors d'un INSERT, le fichier intermédiaire peut déjà contenir des enregistrements du type ″70″, de même identifiant, provenant de précédentes transactions. Les nouveaux enregistrements sont insérés à la suite.

CALPUF

Cet ordre se transforme en appel au sous-programme PUF. Cela déclenchera la mise à jour de la base à partir des enregistrements précédemments stockés dans le fichier de travail, à l'aide de l'ordre INSERT.

FETCHER

A la rencontre d'un ordre FETCHER, l'extracteur restitue une à une les erreurs qui ont pu être stockées dans le fichier intermédiaire par le sous-programme PUF dans la zone pref-PUFERR de pref-PAFCOM. C'est au programme utilisateur de prévoir la boucle de restitution.

Les ordres INSERT, CALPUF et FETCHER sont indépendants des curseurs et n'existent qu'en TP.

LES PARAMETRES DU TRANSLATEUR ET L'ORDRE SET

SET

Cet ordre permet la modification dynamique de certains paramètres pour le fonctionnement du translateur. Pour tout programme utilisateur translaté, ces paramètres sont initialisés par la deuxième ligne (en commentaire, après IDENTIFICATION DIVISION, voir le sous-shapitre "Fonctionnement général").

A la rencontre de l'ordre SET, le translateur, après avoir mis à jour les paramètres précisés, reporte dans le programme translaté l'ordre en commentaire et ne génère rien de plus. SET est donc le seul ordre PAF à ne pas être traduit par des phrases COBOL.

Pour générer les zones nécessaires à PUF, vous devez déclarer l'ordre SET au tout début de la Working Storage Section, dans le premier EXEC PAF. C'est dans cet EXEC PAF SET que vous pouvez utiliser les paramètres UPDATE, BASPUF et IDPUF, qui sont spécifiques à PUF.

Vous pouvez aussi utiliser l'ordre SET dans la Procedure Division si vous l'avez d'abord déclaré au tout début de la Working Storage Section, dans le premier EXEC PAF. Vous pouvez dans ce cas l'utiliser pour les paramètres SESSION, LIB... qui sont modifiés dynamiquement.

Syntaxe :

```
EXEC PAF  SET
    LIB = <code-bibliothèque>
    DELIM  = <délimiteur>
    TYPE = <variantes-génération>
    MODE = <mode-exécution>
    LINK = <mode-d'appel>
    SESSION = <session-et-version>
    USRENT = <EU-param>
    CTXCOL = <CTX-param>
    UPDATE = <pref>
    BASPUF = <base-PUF>
    IDPUF = <ident-PUF>
END-EXEC
```

Cette syntaxe est du même type que celle de l'ordre CONNECT. En particulier, il n'est pas obligatoire d'affecter une nouvelle valeur à l'ensemble des paramètres LIB, DELIM, TYPE et MODE, ni de les écrire dans cet ordre.

LES PARAMETRES DU TRANSLATEUR

Les paramètres situés à la droite du signe = et décrits ci-dessous ne sont pas, contrairement au cas du CONNECT, des constantes ou des variables COBOL. Les paramètres sont donnés par leur valeur alphanumérique pure, sans délimiteur de chaînes. Par exemple, le code Bibliothèque étant une chaîne de 3 caractères, il faut écrire pour modifier ce dernier :

EXEC PAF SET LIB = T00 END-EXEC

Description des paramètres :

LIB : (3 caractères)

Permet de modifier la Bibliothèque (par défaut, c'est la Bibliothèque de génération). Ceci est utile pour déclarer des curseurs sur des tables d'EU lorsque les Méta-Entités correspondantes ne sont pas définies dans la bibliothèque de génération.

DELIM : (6 caractères)

Permet de modifier le délimiteur (DELIM = SINGLE pour la quote ' ou DELIM = DOUBLE pour la double quote ").

TYPE : (2 caractères)

Permet de modifier le type de COBOL à générer en terme de variante / sous-variante VA Pac. Ceci est utile par exemple pour remplacer IBM-CICS (00 ou X0) par IBM-IMS (01 ou X1) quand la distinction n'existe pas (génération batch), les appels de sous-programmes accédant à des fichiers étant spécifiques en IMS. Voir les valeurs de ces variantes dans les Manuels "Applications Batch" et "Dialogue'.

MODE : (5 caractères)

Permet de modifier le mode d'exécution (batch ou TP), pour ceux utilisant le générateur batch pour écrire des programmes TP.

LINK : (6 caractères)

Permet de spécifier le mode d'appel de l'extracteur (STATIC ou DYNAM) respectivement pour des appels statiques ou dynamiques. Pour CICS, ce paramètre n'est pas pris en compte et le translateur génère un ordre LINK pour l'appel à l'extracteur.

SESSION : (5 caractères)

Permet de modifier la session (session de génération VisualAge Pacbase par défaut).

USRENT : (1 à 5 caractères)

Permet de modifier le code de la colonne ″mots-clés″ de la table de définitions d'Entité Utilisateur afin d'éviter un conflit avec une Rubrique Utilisateur utilisée dans la Description de la Meta-Entité associée. Cette colonne est en standard WOEU et désigne les mots-clés explicites dans cette table. Le code identifiant de la fiche (trouvée sur le -CE00) est repris en tête de ces descriptions.

Outre celle-ci, chaque table $TTDxx est composée de colonnes spécifiques correspondant aux Rubriques appelées dans les lignes -CExx de la Meta-Entité associée Le paramètre USRENT se substitue à la chaîne ″OEU″ dans le code colonne:

USRENT = XXX renomme cette colonne WXXX.

CTXCOL : (1 à 5 caractères)

Permet de modifier les codes des colonnes de contexte de toutes les tables PAF afin d'éviter les conflits dans une table de définition ou de description d'EU avec des Rubriques utilisateur utilisées dans la description de la Méta Entité associée. Ces colonnes sont en standard LCTX, SCTX, TCTX et NCTX et désignent respectivement le code bibliothèque, la session, la version dans la session et le sous-réseau (voir le sous-chapitre ″Le Programme Utilisateur TRANSLATE″). Le paramètre CTXCOL se substitue à la chaîne ″CTX″ dans chacun de ces codes colonnes :

CTXCOL = YYY renomme respectivement ces colonnes LYYY, SYYY, TYYY et NYYY.

Il faut cependant s'assurer que les paramètres USRENT et CTXCOL sont bien distincts afin d'éviter d'autres conflits.

La clause ″UPDATE = pref″

Cette clause génère la description de deux workings qui seront utilisées lors des appels à l'extracteur PAF (pref-PAFCOM) ou au sous-programme PUF (pref-PUFCOM) à l'aide des ordres INSERT, CALPUF et FETCHER - ″pref″ étant un préfixe de 1 à 4 caractères -. Cette clause ne peut être utilisée que dans la Working Storage Section, dans l'ordre SET.

Les clauses ″BASPUF = base-PUF″ et ″IDPUF = ident-PUF″

Les ordres propres à PUF étant indépendants des curseurs PAF, il faut déclarer des identifiants spécifiques à PUF pour permettre l'accès au fichier intermédiaire de travail. - ″base-PUF″ : code transaction de la base VA Pac (4 caractères) - ″ident-PUF″ : identifiant de 1 à 25 caractères.

Ces deux clauses ne peuvent être utilisées que dans la Working Storage Section, dans l'ordre SET.

Pour ces 2 dernières clauses, les paramètres situés à la droite du signe = peuvent être des constantes (à renseigner entre quotes) ou des variables COBOL (cas du CONNECT).

Remarque : L'ordre SET ne permet pas de modifier le préfixe des programmes appelés; en effet, la valeur est fixe et est égale à 'BVP'.

INITIALISATIONS ET TERMINAISONS TECHNOLOGIQUES

Les ordres INIT et QUIT permettent à l'extracteur d'effectuer respectivement les initialisations et terminaisons technologiques, dépendant du système d'exploitation. Les ordres INIT et QUIT doivent donc être exécutés respectivement avant et après tout autre ordre PAF. Ils ne sont obligatoires qu'en batch.

INIT

EXEC PAF INIT END-EXEC

QUIT

EXEC PAF QUIT END-EXEC

REMARQUE : INIT et QUIT en Batch

Si le programme utilisateur PAF fait appel à des sous-programmes, il est possible de n'exécuter ces ordres qu'une seule fois (dans le programme appelant) mais nécessaire de les écrire dans chaque sous-programme contenant des ordres PAF.

Optimisation des accès à la Base

INTERET DE LA CLAUSE WHERE

L'utilisation de la clause WHERE permet d'améliorer les performances à deux niveaux ; optimisation des accès à la Base et réduction du volume des données extraites stockées dans le fichier intermédiaire SYSPAF.

- Optimisation des accès à la Base :

L'accès à une table sera plus ou moins performant selon l'utilisation de la clause WHERE.

EXEMPLE : Accès à la Table SEGRUB.

- SELECT * FROM SEGRUB WHERE CRUB = 'cccccc'

  Index des chaînages de la Rubrique sur les Segments

  Choix équivalent : EccccccXS
- SELECT * FROM SEGRUB WHERE CSEG = 'ssss'

  Accès direct à la Description du Segment

  Choix équivalent : SssssCE
- SELECT * FROM SEGRUB WHERE CSEG > 'ssss'

  Accès préalable à la Liste des Segments, puis pour chaque Segment on accède à ses appels de Rubriques

  Choix équivalents : LCSssss, S____CE
- Réduction du volume des données stockées dans le fichier intermédiaire (SYSPAF) :

  Deux démarches sont possibles :

  – Soit sélectionner tous les Programmes pour ne retenir ensuite que les Programmes dont la Colonne TPGMNA = M,

  – Soit sélectionner dans une clause WHERE uniquement les Programmes dont le code colonne TPGMNA = M.

    L'extraction sera bien évidemment plus rapide si l'utilisateur opte pour la seconde solution.

CONCLUSION : Il est donc vivement conseillé d'utiliser la clause WHERE et de lui donner le maximum de critères restrictifs.

INTERET DU PARAMETRE SIZE DANS L'ORDRE CONNECT

Afin d'optimiser les accès à la base VisualAge Pacbase, l'extracteur permet de lire plusieurs enregistrements en avance. L'intérêt de ces lectures en avance est de ne pas être pénalisé par une reprise de lecture à chaque fois qu'on fetche un enregistrement. En effet, ces lectures provoquent un repositionnement systématique (START) sur le fichier des index (AN).

Ces lectures en avance peuvent donc être vues comme l'équivalent logique des BUFFERs d'entrée-sortie utilisés par les méthodes d'accès aux fichiers dans tout système d'exploitation.

Pour chaque curseur, PAF autorise le paramétrage de ce nombre de lectures en avance par le paramètre SIZE de l'ordre CONNECT.

GESTION DES ENREGISTREMENTS LUS PAR L'EXTRACTEUR

Pour mieux comprendre la façon d'optimiser les accès à la base, il est nécessaire de détailler le comportement de l'extracteur en ce qui concerne les lectures en avance. Deux ordres peuvent provoquer des lectures de la base : OPEN et FETCH.

La fonction de lecture de l'extracteur consiste à lire la base, à vérifier la validité de l'enregistrement lu et à le stocker dans le fichier de travail s'il est valide. Ce cycle est répété tant que le nombre d'enregistrements stockés est inférieur au nombre donné par le paramètre SIZE.

Le traitement peut cependant s'arrêter si la fonction de lecture détecte la fin des enregistrements pour le curseur. Dans tous les cas, la fonction de lecture retourne le nombre d'enregistrements effectivement lus et un indicateur de fin de curseur.

Cette fonction de lecture est systématiquement exécutée à l'ouverture du curseur. Elle peut également l'être lors du traitement d'un ordre FETCH. Le fetching consiste essentiellement à lire le fichier de travail pour restituer un nouvel enregistrement. Dans le cas où l'enregistrement restitué est le dernier lu par la fonction de lecture, celle-ci est à nouveau exécutée si elle n'a pas précédemment détecté la fin du curseur.

OPTIMISATION PAR LE PARAMETRE SIZE

Le paramètre SIZE est donc essentiel dans l'efficacité de l'extracteur PAF en terme de temps de réponse.

Ainsi, en TP, il est pénalisant de prendre un SIZE inférieur au nombre d'enregistrements fetchés par écran (nombre d'enregistrements affichés sur l'écran par exemple). Dans l'hypothèse simplificatrice d'un curseur non sélectif (tous les enregistrements lus sont valides), le SIZE optimal est un multiple du nombre d'enregistrements fetchés par écran.

Dès lors que la conversation peut être interrompue à chaque réception d'écran, le SIZE optimal est effectivement égal au nombre d'enregistrements fetchés par écran.

En batch, le problème doit être pris en sens inverse. Dans le cas d'un curseur qu'on doit fetcher jusqu'à son terme, il semble intéressant, à première vue, de choisir un paramètre SIZE très grand pour minimiser le nombre de reprises de lecture. Cependant, des valeurs de SIZE trop importantes provoquent la création dynamique d'une quantité d'enregistrements pénalisante pour la plupart des méthodes d'accès aux fichiers séquentiels indexés.

L'idéal en batch est de définir une taille de BUFFER d'entrée-sortie pour le fichier de travail suffisante pour qu'une exécution de la fonction de lecture ne provoque que des entrées-sorties logiques.

Ces quelques remarques devraient permettre aux programmes utilisant PAF de fonctionner avec des temps d'exécution ou des temps de réponse satisfaisants.

Recommandation pour le paramètre IDENT

PRINCIPE GENERAL

Le principe général pour l'alimentation du paramètre IDENT est d'identifier de façon unique une conversation en environnement multi-utilisateurs. Cette identification est donc naturellement dépendante en grande partie du moniteur TP sous lequel s'exécute le programme translaté.

Lorsque cela est possible, les recommandations ci-dessous tiennent compte des variables standard fournies par le moniteur TP (identifiant du terminal, etc...).

OS390-CICS

La variable CICS EIBTRMID identifie chaque terminal.

IMS

Le nom du terminal logique se trouve dans la zone IO/PCB (variable S-IPCB-XNMTE dans DIALOGUE).

Mise en oeuvre sous VisualAge Pacbase

La clause DECLARE CURSOR doit figurer en WORKING STORAGE SECTION. Elle sera donc insérée complètement sur des lignes ″-W″ du Programme ou de l'Ecran, à un emplacement choisi par l'utilisateur. Dans les lignes -W, il convient d'écrire la chaîne 'EXEC PAF' en colonne 5 et la chaîne 'END-EXEC' après la phrase de déclaration du curseur.

Exemple

```
NLG S NIVEAU          DESCRIPTION
```

```
100       EXEC PAF DECLARE CU01 CURSOR FOR
110       SELECT * FROM RUBDEF
120       WHERE FRUBLI = 5 END-EXEC
```

Exceptée la clause DECLARE CURSOR, tous les ordres SQL s'insèrent dans le programme utilisateur par le langage structuré. Un opérateur spécifique (EXP) permet, sur ces lignes, de générer les appels EXEC PAF et END-EXEC, qui encadrent les ordres SQL-PAF.

Exemple

```
OPE  OPERANDE
EXP  OPEN CU01
```

génère :

```
EXEC PAF
OPEN CU01
END-EXEC
```

Le Programme Utilisateur "translaté"

En amont du compilateur COBOL, le translateur PAF traduit les requêtes SQL-PAF en déclarations ou instructions COBOL. (Attention : le translateur PAF est incompatible avec une demande de formatage COBOL.) Les séquences EXEC PAF ... END-EXEC sont reportées dans le programme COBOL sous forme de commentaires.

WORKING STORAGE SECTION

La phrase 'DECLARE <code-curseur> CURSOR FOR SELECT * FROM <code-table>' génère, en WORKING STORAGE SECTION du programme utilisateur, sous le niveau 01 <code-curseur>-CURSOR, les déclarations de données suivantes :

- la zone de gestion du curseur,
- la zone de spécifications, où la requête est traduite,
- la zone de communication, description de la table PAF sélectionnée.

Dans cette génération, les zones accessibles dans le programme utilisateur (gestion du curseur et communication) sont préfixées par le code du curseur.

La zone de spécifications est générée sous forme d'un FILLER.

Exemple

Extraction de descriptions de Textes (table TXTDSC), pour le Texte de code TEXT01 et de code division supérieur à EE. Soit le curseur de code TX04 :

```
EXEC PAF  DECLARE TX04 CURSOR FOR SELECT * FROM
          TXTDSC WHERE CTXT = 'TEXT01' AND CDIV > 'EE'
END EXEC
```

Les zones suivantes sont générées sous le niveau :

```
   01    TX04-CURSOR.
```

ZONE DE GESTION DU CURSEUR

En batch :

```
05      TX04-SAVE.
  10    FILLER        PIC X(06) VALUE 'TX0401'.
  10    TX04-CODTAB   PIC X(10) VALUE 'TXTDSC    '.
  10    TX04-CODRET   PIC 9(00002).
  10    TX04-ORDRE    PIC X(00001).
  10    TX04-FI       PIC X(00001).
  10    TX04-FT       PIC X(00001).
  10    TX04-CUTICU   PIC X(00008).
  10    TX04-CPASCU   PIC X(00008).
  10    TX04-CBIBCU   PIC X(00003).
  10    TX04-CSESCU   PIC X(00005).
  10    TX04-CRESCU   PIC X(00001).
  10    TX04-NENRCU   PIC 9(00006).
  10    TX04-INTERN   PIC X(00034).
```

où

CODTAB : Code de la table.

CODRET : Code retour de l'extracteur.

Si aucune erreur n'a été décelée, le code retour est à zéro.

Se reporter au chapitre ″Messages d'erreurs″, sous-chapitre ″Extracteur″, pour une description exhaustive de ces codes retour.

ORDRE : Chaque ordre PAF est repéré par un numéro :

1 : INIT

2 : CONNECT

4 : OPEN

6 : FETCH

8 : CLOSE

9 : QUIT

FI : Fin de lecture du curseur :

1 : lecture du dernier enregistrement du curseur

0 : sinon

FT : Fin de traitement du curseur :

1 : FETCH d'un curseur au-delà de son dernier enregistrement

0 : sinon

NOTE : FI est positionné à '1' dans deux cas :

- L'ordre FETCH retourne le dernier enregistrement correspondant à la sélection du curseur,
- Aucune donnée n'est sélectionnée par l'ordre OPEN.

  Si on exécute un ordre FETCH quand FI = '1', l'indicateur FT est retourné à '1' sans nouvel enregistrement exploitable.

  L'ordre OPEN peut donc lui aussi renseigner l'indicateur FI pour éviter un FETCH inutile.

Les zones suivantes sont les paramètres utilisateur de l'ordre CONNECT ; leur alimentation est donc générée par le translateur, en PROCEDURE DIVISION, à la rencontre de cet ordre.

CUTICU : Code utilisateur VisualAge Pacbase.

CPASCU : Mot de passe VisualAge Pacbase.

CBIBCU : Code de la Bibliothèque de connexion.

CSESCU : Session de connexion et version de la session.

CRESCU : Option de sélection du sous-réseau.

NENRCU : Nombre maximum d'enregistrements du fichier intermédiaire.

INTERN : Zone technologique interne.

NOTE : La zone de gestion du curseur est intitulée <code-curseur>-SAVE car, en TP, c'est la zone qu'il convient de sauvegarder lorsque le programme appelant rend la main au moniteur. L'extracteur TP

sauvegarde en effet une représentation de la requête (zone <code-curseur>-TECH, voir ci-dessous) dans le fichier de travail.

En TP :

La zone de gestion est la même qu'en batch, mais avec deux zones supplémentaires positionnées devant.

Elles permettent d'identifier la base et le terminal (voir la description de l'ordre CONNECT) :

```
10    TX04-IDENT      PIC X(25).
10    TX04-BASE       PIC X(4).
```

Dans tous les cas, il est impératif de ne pas modifier le contenu des zones -SAVE et -TECH (voir ci-dessous).

ZONE DE SPECIFICATIONS

Déclarée sous le niveau : 05 TX04-TECH, cette zone est un FILLER de longueur variable, où la requête est traduite pour l'extracteur. L'utilisateur n'y a pas accès.

ZONE DE COMMUNICATION

```
05      TX04.
  10    TX04-LCTX     PIC X(00003).
  10    TX04-SCTX     PIC 9(00004).
  10    TX04-TCTX     PIC X(00001).
  10    TX04-NCTX     PIC X(00001).
  10    TX04-CTXT     PIC X(00006).
  10    TX04-CDIV     PIC X(00002).
  10    TX04-CLIG     PIC 9(00003).
  10    TX04-TLIG     PIC X(00001).
  10    TX04-DLIGTX   PIC X(00060).
  10    TX04-CRUB     PIC X(00006).
```

Quelle que soit la table sélectionnée, les quatre premières colonnes sont toujours générées. Elles précisent l'origine de la ligne extraite, par rapport au contexte de connexion de l'utilisateur :

LCTX : Code Bibliothèque de définition de la ligne.

SCTX : Numéro de session de dernière modification de la ligne.

TCTX : Etat de la session de dernière modification de la ligne.

NCTX : Témoin de provenance de la ligne, par rapport au contexte de connexion du curseur :

= : la ligne est issue de la Bibliothèque de connexion,

> : la ligne est issue d'une Bibliothèque hiérarchiquement supérieure à la Bibliothèque de connexion,

< : la ligne est issue d'une Bibliothèque hiérarchiquement inférieure à la Bibliothèque de connexion.

Les codes de ces quatre colonnes peuvent être modifiés par l'ordre SET grace au paramètre CTXCOL.

Les autres colonnes sont les colonnes spécifiques de la table sélectionnée.

PROCEDURE DIVISION

En PROCEDURE DIVISION, chaque ordre SQL est remplacé par :

- L'alimentation de la zone TX04-ORDRE,
- L'appel au sous-programme d'extraction, auquel est transmis toute la zone 01 TX04-CURSOR.

A la rencontre d'un ordre CONNECT, le translateur génère de plus l'alimentation des zones de gestion du curseur, qui décrivent les paramètres utilisateur.

A la rencontre d'un ordre OPEN, le translateur génère également l'alimentation des zones techniques du curseur (-TECH) lorsque ce dernier dépend d'une ou de plusieurs zones COBOL. Ces alimentations sont dans certains cas conditionnées.

Imbrication des curseurs PAF

LIMITATION DE LA SYNTAXE PAF-SQL

La syntaxe PAF-SQL ne retient qu'un sous-ensemble restreint du langage SQL. En particulier, il n'est pas possible de définir de curseurs avec des clauses SELECT imbriquées. L'imbrication des clauses SELECT est cependant utile pour obtenir des informations conditionnées par des séquences de chaînages par exemple (liste des Rubriques appelées dans les Segments utilisés dans les Programmes...).

Cependant, cette limitation n'est pas réellement contraignante dès lors que PAF autorise la déclaration de plusieurs curseurs (jusqu'à 100), ces curseurs pouvant dépendre de zones COBOL. En effet, si un curseur dépend d'une

zone COBOL faisant partie de la zone de communication d'un autre curseur, les deux curseurs équivalent à un curseur utilisant des clauses SELECT imbriquées.

IMBRICATION DES CURSEURS

Considérons les tables SDODEF (Fiche Structure de Données) et PGMSDO (Appel de SD dans un Programme). Considérons d'autre part les colonnes CSDO (Code SD) et CPGM (Code Programme). Pour obtenir la liste des Programmes appelant la SD pour toutes les SD de la base, la clause SELECT de la requête SQL standard s'écrit :

SELECT * FROM PGMSDO WHERE

CSDO = (SELECT CSDO FROM SDODEF)

Il est possible de simuler cette imbrication en SQL-PAF par le jeu de 2 curseurs :

DECLARE LCD CURSOR FOR SELECT * FROM SDODEF

DECLARE PGCD CURSOR FOR SELECT * FROM PGMSDO WHERE CDSO = LCD-CSDO

On constate que le premier curseur permet d'obtenir la liste de toutes les SD. Le second permet d'obtenir tous les Programmes appelant la SD dont le code est LCD-CSDO, c'est-à-dire le code de la SD actuellement fetchée dans le premier curseur.

En PROCEDURE DIVISION, après connexion des deux curseurs, on ouvre le curseur LCD. Puis, à chaque FETCH sur LCD et tant que LCD-FT n'est pas à 1 (curseur non terminé), on ouvre le curseur PGCD. Cette ouverture permet la prise en compte du nouveau code SD dans la zone LCD-CSDO (voir le sous-chapitre "Le Programme Utilisateur translaté"). Le curseur PGCD est alors fetché jusqu'à son terme, puis fermé, et on retourne au fetching du curseur LCD.

Ainsi, on simule par imbrication des traitements de curseurs un curseur unique défini par des clauses SELECT imbriquées. Notons que ce mécanisme peut se généraliser à une profondeur quelconque d'imbrications dans la limite de 100 curseurs au maximum pour un programme utilisateur.

Exécution du Programme Utilisateur

Batch

En batch, le fichier intermédiaire de stockage est défini pour un job. Il est donc nécessaire, en début de chaîne, de créer le fichier (et éventuellement de détruire le précédent). Ce fichier est un fichier séquentiel indexé avec une clé de longueur 12 et une taille maximum d'enregistrement égale à 1031, la taille moyenne étant 170.

Ensuite, pour exécuter son programme batch l'utilisateur doit, dans son JCL, déclarer les fichiers nécessaires à l'exécution de l'extracteur (fichiers VisualAge Pacbase et fichier intermédiaire de stockage).

TP

En TP, le fichier intermédiaire de stockage est un séquentiel indexé avec une clé de longueur 37 commençant en position 2 et une taille maximum d'enregistrement égale à 1161, la taille moyenne étant 200.

Chapitre 3. Exemples de Programmes utilisant PAF

Introduction

Ce chapitre présente deux exemples d'utilisation de PAF permettant de présenter la mise en oeuvre de PAF en batch et en TP et de proposer des utilisations types de PAF (contrôle de qualité de normes, administration de la base, ...).

Le premier programme, PAFEX1, fonctionne en mode batch. Il dresse la liste de toutes les Propriétés (Rubriques de type 'P') qui n'ont pas de nom relationnel. Il nécessite la déclaration de deux curseurs : un pour la définition des Rubriques et un pour les descriptions. Ce programme constitue donc un bon exemple d'utilisation de curseurs imbriqués.

Le second programme, PAFEX2, est un dialogue dressant la liste des écrans ne se conformant pas à un standard local (présentation des Rubriques, caractère d'initialisation, demandes d'aide Ecran ou Rubrique). Ce programme utilise un seul curseur (pour les définitions d'Ecran) et gère les défilements d'écrans. Il illustre le mécanisme de transmission du contexte PAF entre chaque itération du dialogue.

Exemple Batch

Objectif :

- Dresser la liste des Propriétés n'ayant pas de nom relationnel.

Déclarations PAF :

- CU01 sélectionne les Rubriques de type 'P'.
- CU02, ouvert pour chaque Rubrique trouvée par CU01, sélectionne les lignes de description contenant un nom relationnel.

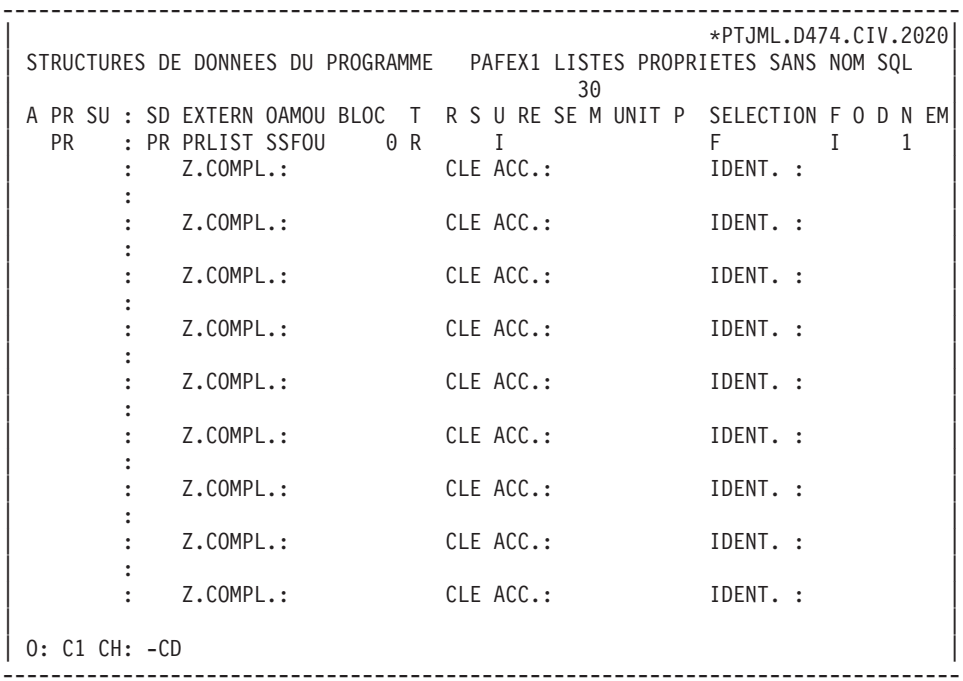
Logique procédurale :

- F02BA : Initialisation de PAF.
- F02CA : Connexion du curseur CU01. Le code utilisateur et le mot de passe sont ici "en dur" mais pourraient être obtenus par lecture d'un fichier en entrée. Il faut noter que le sous-réseau est fixé à 'U', ce qui signifie que seules les Propriétés de la Bibliothèque CIV sont prises en compte. L'ordre CONNECT établit le contexte de la lecture PAF.
- F02DA : Connexion du curseur CU02.
- F21BA : Ouverture du curseur CU01. Cet ordre se traduit par la lecture des fiches de définition des Rubriques de type 'P' de la bibliothèque CIV et par l'écriture de ces fiches dans le fichier de travail PAF.

- F21CA : Ces fiches de définitions sont appelées, c'est-à-dire relues une à une à partir du fichier de travail PAF (ordre FETCH).
- F21DA : Tant que la fin du curseur CU01 n'est pas atteinte (CU01-FI = '0'), le curseur CU02 est ouvert pour chaque Rubrique lue. Ce curseur sélectionne uniquement les lignes de type 'R' dans les descriptions de la Rubrique lue. Par conséquent, une fin de curseur immédiate (CU02-FI = '1') dès le premier ordre FETCH signifie que cette Propriété n'a pas de nom relationnel.

  Lorsque cette condition se produit, une ligne est formattée et imprimée sur un Etat. Le curseur CU02 est finalement fermé pour permettre sa réouverture avec la Rubrique suivante du curseur CU01.
- F79 : Lorsque toutes les Propriétés ont été lues par l'ordre FETCH, le curseur CU01 est fermé et un ordre QUIT est exécuté afin de fermer les fichiers de la base et le fichier de travail PAF.

```
                                                                    *PTJML.D474.CIV.2020
FICHE DU PROGRAMME        PAFEX1

NOM DU PROGRAMME...................: LISTES PROPRIETES SANS NOM SQL

CODE CLASSEMENT DU PROGRAMME.......: PAFEX1

VARIANTE DU LANGAGE A GENERER......: 0
OPTION NUMEROTATION CADRAGE COBOL..:
OPTION CARTES AVANT PROGRAMME......:
OPTION CARTES APRES PROGRAMME......:
CODE DU PROGRAMME GENERE...........: PAFEX1
TYPE DE PROGRAMMATION..............: P
NATURE DU PROGRAMME................: B
TYPE DE L'ENTITE...................: P   PROGRAMME
CONTROLE DE PRESENCE ZONE NUMERIQUE:



MOTS CLES ASSOCIES.:
MIS A JOUR PAR.....:            LE :            A :   :  :      BIB :
NO DE SESSION......: 2013       BIBLIOTHEQUE : CIV    BLOCAGE :

O: C1 CH: Ppafex1                    ACTION:
```


*PTJML.D474.CIV.2020
STRUCTURES DE DONNEES DU PROGRAMME PAFEX1 LISTES PROPRIETES SANS NOM SQL
30
A PR SU : SD EXTERN OAMOU BLOC T R S U RE SE M UNIT P SELECTION F O D N EM
PR : PR PRLIST SSFOU 0 R I F I 1
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
: Z.COMPL.: CLE ACC.: IDENT. :
O: C1 CH: -CD

```
                                                                        *PTJML.D474.CIV.2020
ZONES DE TRAVAIL DU PROGRAMME   P PAFEX1 LISTES PROPRIETES SANS NOM SQL

DEBUT DU NUMERO DE LIGNE : BA
A NLG S NIVEAU           DESCRIPTION                                        TABLE
  100 * LISTE DES PROPRIETES
  110       EXEC PAF DECLARE CU01 CURSOR FOR
  120       SELECT * FROM RUBDEF WHERE
  130       TRUB = 'P'
  140       END-EXEC
  200 * LISTE DES NOMS RELATIONNELS D'UNE PROPRIETE
  210       EXEC PAF DECLARE CU02 CURSOR FOR
  220       SELECT * FROM RUBDSC WHERE
  230       CRUB = CU01-CRUB AND
  240       TLIG = 'R'
  250       END-EXEC









O: C1 CH: -W
```

```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS PROGRAMME P PAFEX1 LISTES PROPRIETES SANS NOM SQL FONCTION: 02

A SS NLG OPE OPERANDE                          NVTY CONDITION
         N   INITIALISATION ET CONNEXIONS      05BL
- -- --- --- -------------------------------- ---- --------------------------
  BA     N   INITIALISATION                    10BL
  BA 100 EXP INIT
- -- --- --- -------------------------------- ---- --------------------------
  CA     N   CONNEXION DE CU01                 10BL
  CA 100 EXP CONNECT CU01 TO
  CA 110     USER    = 'USER'
  CA 120     PASS    = 'PASS'
  CA 130     LIB     = 'CIV'
  CA 140     NET     = 'U'
  CA 150     SESSION = SPACES
  CA 160     SIZE    = 40
- -- --- --- -------------------------------- ---- --------------------------
  DA     N   CONNEXION DE CU02                 10BL
  DA 100 EXP CONNECT CU02 TO
  DA 110     USER    = 'USER'
  DA 120     PASS    = 'PASS'

O: C1 CH: -P02
```

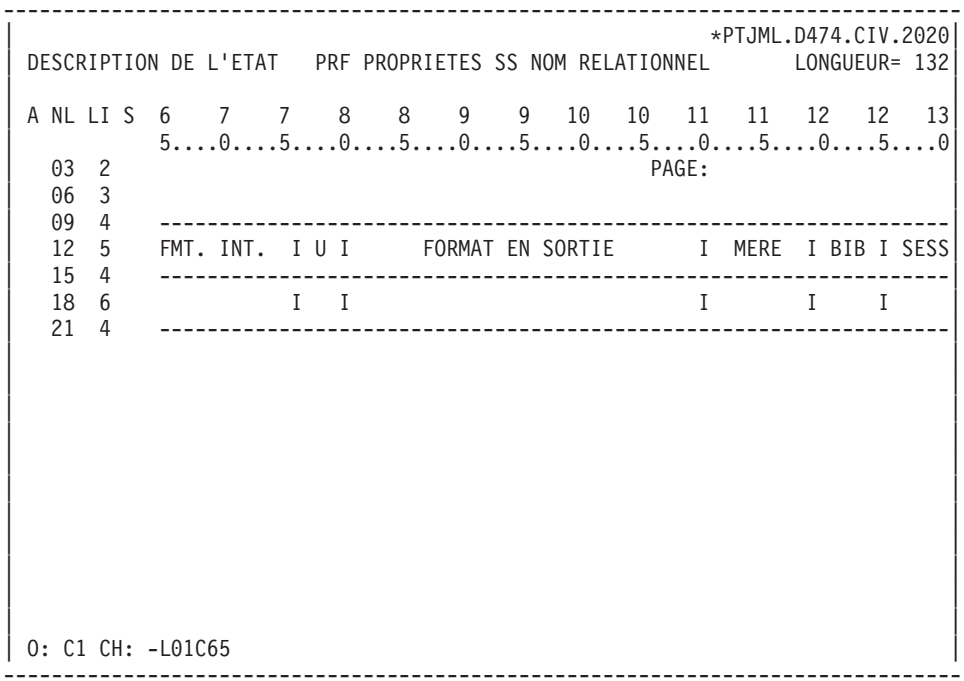
```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS PROGRAMME P PAFEX1 LISTES PROPRIETES SANS NOM SQL FONCTION: 02

A SS NLG OPE OPERANDE                          NVTY CONDITION
  DA 130     LIB     = 'CIV'
  DA 140     NET     = 'U'
  DA 150     SESSION = SPACES
  DA 160     SIZE    = 1
- -- --- --- ------------------------------- ---- -------------------------









O: C1 CH:
```

```
                                                                        *PTJML.D474.CIV.2020
 TRAITEMENTS PROGRAMME P PAFEX1 LISTES PROPRIETES SANS NOM SQL FONCTION: 21

 A SS NLG OPE OPERANDE                          NVTY CONDITION
          N   RECHERCHE                         05BL
 - -- --- --- ------------------------------- ---- --------------------------
   BA     N   OUVERTURE LISTE DES PROPRIETES   10BL
   BA 100 EXP OPEN CU01
 - -- --- --- ------------------------------- ---- --------------------------
   CA     N   LECTURE DES PROPRIETES UNE A UNE 10DW CU01-FI = '0'
   CA 100 EXP FETCH CU01
 - -- --- --- ------------------------------- ---- --------------------------
   DA     N   RECHERCHE DES NOMS RELATIONNELS  15BL
   DA 100 *   OUVERTURE DES NOMS RELATIONNELS
   DA 110 EXP OPEN CU02
   DA 200 *   LECTURE DES NOMS RELATIONNELS
   DA 210 EXP FETCH CU02
   DA 300 *   PAS DE NOM RELATIONNEL : EDITION 99IT CU02-FT = '1'
   DA 310 P   F8F
   DA 400 *   FERMETURE DES NOMS RELATIONNELS  99BL
   DA 410 EXP CLOSE CU02
 - -- --- --- ------------------------------- ---- --------------------------

 O: C1 CH: -P21
```

```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS PROGRAMME P PAFEX1 LISTES PROPRIETES SANS NOM SQL FONCTION: 79

A SS NLG OPE OPERANDE                          NVTY CONDITION
         N   DECONNEXION DE PAF                05BL
     100 EXP CLOSE CU01
     110 EXP QUIT
     120 GFT
- -- --- --- ------------------------------- ---- -------------------------









*** FIN ***
O: C1 CH: -P79
```

```
                                                                        *PTJML.D474.CIV.2020
CODE ETAT               PRF

NOM...................: PROPRIETES SS NOM RELATIONNEL

COMMENTAIRE...........:

NATURE................: E ETAT
TYPE DE L'ETAT........: L

LONGUEUR DU LIBELLE...: 132
FORMAT TOTALISATION   : PARTIE ENTIERE : 11
                        PARTIE DECIMALE: 07




MOTS CLES ASSOCIES.:


MIS A JOUR PAR.....:            LE :            A :   :         BIB :
NO DE SESSION......: 2013       BIBLIOTHEQUE : CIV    BLOCAGE :

O: C1 CH: Rprf                       ACTION:
```

```
                                                                        *PTJML.D474.CIV.2020
DESCRIPTION DE L'ETAT   PRF PROPRIETES SS NOM RELATIONNEL       LONGUEUR= 132

A NL LI S          1    1    2    2    3    3    4    4    5    5    6    6
           1...5....0....5....0....5....0....5....0....5....0....5....0....5.
  00  1
  03  2                         LISTE DES PROPRIETES SANS NOM RELATIONNEL
  06  3                         ----------------------------------------
  09  4    ------------------------------------------------------------------
  12  5    I  CODE  I         NOM DE LA PROPRIETE          I FMT. ENTREE I FM
  15  4    ------------------------------------------------------------------
  18  6    I        I                                      I             I
  21  4    ------------------------------------------------------------------

O: C1 CH: -L
```


*PTJML.D474.CIV.2020
DESCRIPTION DE L'ETAT PRF PROPRIETES SS NOM RELATIONNEL LONGUEUR= 132
A NL LI S 6 7 7 8 8 9 9 10 10 11 11 12 12 13
5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0
03 2 PAGE:
06 3
09 4
12 5 FMT. INT. I U I FORMAT EN SORTIE I MERE I BIB I SESS
15 4
18 6 I I I I I
21 4
O: C1 CH: -L01C65

```
                                                                        *PTJML.D474.CIV.2020
COMPOSITION DE L'EDITION  PRF PROPRIETES SS NOM RELATIONNEL

A:   LONGUEUR LIGNE: 132 NB LIGNES: 60 NB.POSTES CAT    OPTION:   SECTION: 00
     COMMENTAIRES:              CONDITION:

A CT NLG T CAR ST LB SAU FOSF COMMENTAIRES  CONDITION
  BA 010        1 01 01*      EN-TETE       5-PR00-FCL NOT < 5-PR00-FCLM
  BA 030          02 01
  BA 050          03 01
  BA 070          04 01
  BA 090          05 01
  BA 110          04 01
- -- --- - --- -- -- --- ---- ------------- ---------------------------------
  CA 010        2 06 01       REPETITIVE
- -- --- - --- -- -- --- ---- ------------- ---------------------------------
  DA 010          04 01       BAS DE PAGE   5-PR00-FCL NOT < 5-PR00-FCLM OR
  DA 020                                    CU01-FI = '1'
- -- --- - --- -- -- --- ---- ------------- ---------------------------------




O: C1 CH: -D
```

```
                                                                    *PTJML.D474.CIV.2020
STRUCTURES DE L'ETAT     PRF PROPRIETES SS NOM RELATIONNEL

A ST RUBRIQ L : DEB S O W FFNNRUBRIQIND CONDITION                          BIBLI
  01 XPAGE  0 : 112   M 5 PR00-FCP                                         2013
- -- ------ - : --- - - - ------------ ------------------------------ -----
  02 CRUB   0 :   3   M   CU01CRUB                                         2013
  02 LRUB   0 :  12   M   CU01LRUB                                         2013
  02 FRUBE  0 :  51   M   CU01FRUBE                                        2013
  02 FRUBI  0 :  65   M   CU01FRUBI                                        2013
  02 ORUBZS 0 :  78   M   CU01ORUBZS                                       2013
  02 FRUBS  0 :  82   M   CU01FRUBS                                        2013
  02 CRUBM  0 : 112   M   CU01CRUBM                                        2013
  02 LCTX   0 : 121   M   CU01LCTX                                         2013
  02 SCTX   0 : 127   M   CU01SCTX                                         2013
  02 TCTX   0 : 131   M   CU01TCTX                                         2013
- -- ------ - : --- - - - ------------ ------------------------------ -----
               :
               :
               :
               :
               :
*** FIN ***
O: C1 CH: -CE
```

Exemple TP

Objectif :

- Dresser la liste des Ecrans ne se conformant pas à un standard local.

Déclarations PAF :

- CU01 sélectionne les fiches de définition des Ecrans dont la présentation des Rubriques, le caractère d'initialisation ou les demandes d'aides Ecran ou Rubrique diffèrent de ceux entrés par l'utilisateur du programme.

Logique procédurale :

- F02 : Initialisation de PAF uniquement au premier passage (EIBCALEN = 0).
- F06CA : Connexion du curseur CU01. Le code utilisateur et le mot de passe sont ici ″en dur″ mais pourraient être entrés dans un menu et passés via la COMMAREA. Il faut noter que le code de la base VA Pac doit être spécifié (D474). De plus, le code du terminal (EIBTRMID) est affecté à l'identifiant PAF pour garantir l'unicité des clés entre conversations dans le fichier PAF.

  Enfin notons que le paramètre SIZE est fixé à 10, ce qui correspond au nombre de lignes répétitives de l'Ecran. Ainsi on ne lit que le nombre

d'Ecrans nécessaires à un affichage afin d'éviter des lectures trop longues pour le TP et inutiles si l'utilisateur quitte le dialogue sans consulter toute la liste.

Dans le cas de ce programme, la fonction F06 est exécutée uniquement au premier passage. Ainsi l'ordre CONNECT et l'ordre OPEN ne sont exécutés qu'une fois.

Ce programme ne prend donc pas en compte des modifications interactives des critères de sélection. Ceci pourrait être réalisé dans le cas d'une modification des champs spécifiques en en-tête de l'Ecran associés au critère de la requête SELECT définissant le curseur CU01.

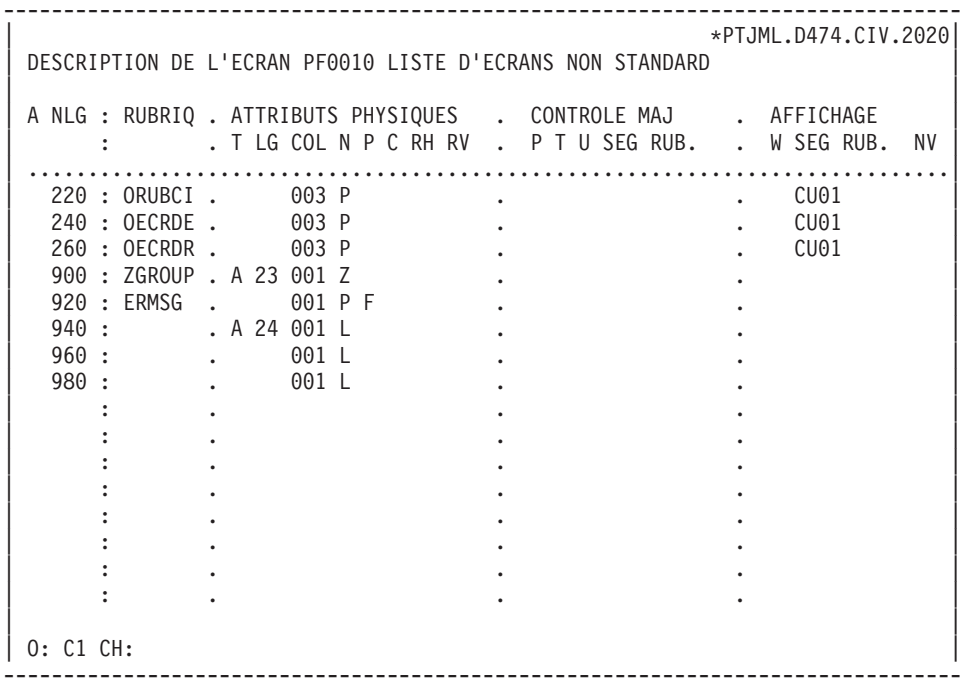
- F06DA : Ouverture du curseur CU01. Cet ordre provoque la lecture des fiches de définition des Ecrans non conformes au standard spécifié dans l'en-tête et le stockage de ces définitions dans le fichier de travail PAF.
- F06EA : La zone de gestion à sauvegarder du curseur CU01 (CU01-SAVE) est transférée dans la COMMAREA par la zone de sauvegarde.
- F4031 : Fermeture du curseur CU01 et terminaison de PAF (CLOSE et QUIT).
- F52BA : Restauration de la zone CU01-SAVE à partir de la COMMAREA. Si tous les enregistrements du curseur ont été lus (ordre FETCH), fermeture du curseur.
- F60PF : Si le dernier enregistrement du curseur n'a pas encore été lu et tant qu'on traite la catégorie répétitive de l'Ecran, on lit l'enregistrement suivant.
- F60PQ : Si le dernier enregistrement a été lu on écrit le message ″FIN″ et on sort de l'itération.
- F75 : S'il n'y a pas eu d'erreur, on sauvegarde le curseur en COMMAREA.

```
                                                                    *PTJML.D474.CIV.2020
COMPLEMENT AU DIALOGUE..............: PF PAF

ZONE COMMUNE DE CONVERSATION.........: PF

FICHIER LIBELLES D'ERREUR
                       ORGANISATION..:
                       NOM EXTERNE...:

PREMIER CODE ECRAN DU DIALOGUE.......:

COMPLEMENT LONGUEUR CONVERSATION.....:

NOM DU PCB OU DU SOUS-SCHEMA.........:


OPTIONS :



N0 DE GENERATION   : 2028        BIBLIOTHEQUE : CIV

O: C1 CH: Opf0010 O                   ACTION:
```

```
                                                                        *PTJML.D474.CIV.2020
DEFINITION DE L'ECRAN .............: PF0010

NOM DE L'ECRAN ...................: LISTE D'ECRANS NON STANDARD

TAILLE DE L'ECRAN (LIGNES,COLONNES):   24       080
PRESENTATION, TABULATION, INITIAL. : * S      * 02
APPEL DE DOC. ECRAN, RUBRIQUE .....: * 11     * 12

                                    LIBELLE  AFFICH. SAISIE L.ERREUR L.ERR.
ATTRIBUT D'INTENSITE ..............:   N        N       N       B       B
ATTRIBUT DE PRESENTATION ..........:   N        N       N       N       N
ATTRIBUT DE COULEUR ...............:   W        W       W       W       W


VARIANTES .........................:   0 * 0   IBM OS CICS (PROG. ET MAP BMS)
CARTES AVANT, CARTES APRES ........: * CC       (PROGRAMME)  * KK      (MAP)
NOMS EXTERNES .....................:   PF001P   (PROGRAMME)    PF001M  (MAP)
TRANSACTION .......................:
MOTS CLES ASSOCIES.:
MIS A JOUR PAR.....:            LE :            A :   :  :      BIB :
NO DE SESSION......: 2013       BIBLIOTHEQUE : CIV    BLOCAGE :

O: C1 CH: Opf0010                    ACTION:
```

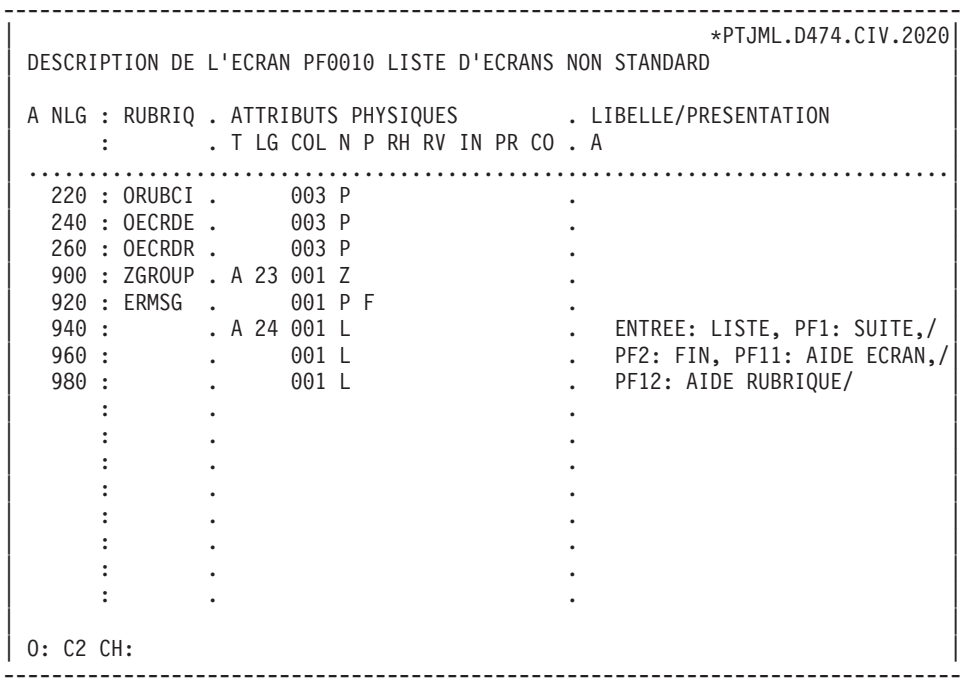
```
                                                                    *PTJML.D474.CIV.2020
 LISTE DES ECRANS UTILISANT L'ECRAN                               PF0010

  ECRAN  :  NLG  RUBRIQ  T LG COL N P C RH RV  C T O SEG RUB.   W SEG RUB.  NV
 ------- APPEL DE TYPE  T  (TITRE) ------------------------------------------
  PF0010    020  PF0010  A 01 025 T     00 00

 O: C1 CH: -XO
```

```
                                                                        *PTJML.D474.CIV.2020
DESCRIPTION DE L'ECRAN PF0010 LISTE D'ECRANS NON STANDARD

A NLG : RUBRIQ . ATTRIBUTS PHYSIQUES   .  CONTROLE MAJ     .  AFFICHAGE
      :        . T LG COL N P C RH RV  .  P T U SEG RUB.   .  W SEG RUB.  NV
.....................................................................................
  010 : PFKEY  .          V            .    O V A          .    EN
  011 :        .                       .      V S          .    01
  012 :        .                       .      G            .    02
  020 : PF0010 . A 01 025 T            .                   .
  025 : PAFVUE .   02 025 V            .  R                .
  040 :        .   02 015 L            .                   .
  045 :        .      001 L            .                   .
  060 : ORUBP1 .   02 001 V            .  R                .
  080 : ORUBC1 .         V            .  P                .
  100 : OECRD1 .   01 001 V            .  P                .
  120 : OECRD2 .         V            .  P                .
  130 :        .   01 001 L            .                   .
  140 : RGROUP .   01 001 R 3      10  .                   .
  160 : CECR   .      003 P            .                   .    CU01
  180 : LECR   .      003 P            .                   .    CU01
  200 : ORUBPL .      003 P            .                   .    CU01

O: C1 CH: -CE
```



```
                                                                        *PTJML.D474.CIV.2020
DESCRIPTION DE L'ECRAN PF0010 LISTE D'ECRANS NON STANDARD

A NLG : RUBRIQ . ATTRIBUTS PHYSIQUES   .  CONTROLE MAJ     .  AFFICHAGE
      :        . T LG COL N P C RH RV  .  P T U SEG RUB.   .  W SEG RUB.  NV
.....................................................................................
  220 : ORUBCI .      003 P            .                    .    CU01
  240 : OECRDE .      003 P            .                    .    CU01
  260 : OECRDR .      003 P            .                    .    CU01
  900 : ZGROUP . A 23 001 Z            .                    .
  920 : ERMSG  .      001 P F          .                    .
  940 :        . A 24 001 L            .                    .
  960 :        .      001 L            .                    .
  980 :        .      001 L            .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .
      :        .                       .                    .

O: C1 CH:
```

```
                                                                        *PTJML.D474.CIV.2020
DESCRIPTION DE L'ECRAN PF0010 LISTE D'ECRANS NON STANDARD

A NLG : RUBRIQ . ATTRIBUTS PHYSIQUES         . LIBELLE/PRESENTATION
      :        . T LG COL N P RH RV IN PR CO . A
.....................................................................
  010 : PFKEY  .          V                  .
  011 :        .                             .
  012 :        .                             .
  020 : PF0010 . A 01 025 T                  .
  025 : PAFVUE .   02 025 V                  . I C

  040 :        .   02 015 L                  .   ENTREZ LES CARACTERISTIQUES/
  045 :        .      001 L                  .   DES ECRANS STANDARD:/
  060 : ORUBP1 .   02 001 V                  . I S
  080 : ORUBC1 .          V                  . I _
  100 : OECRD1 .   01 001 V                  . I 11
  120 : OECRD2 .          V                  . I 12
  130 :        .   01 001 L                  . A 079-
  140 : RGROUP .   01 001 R 3    10          .
  160 : CECR   .      003 P                  .
  180 : LECR   .      003 P                  .
  200 : ORUBPL .      003 P                  .

O: C2 CH: -CE
```


*PTJML.D474.CIV.2020
DESCRIPTION DE L'ECRAN PF0010 LISTE D'ECRANS NON STANDARD
A NLG : RUBRIQ . ATTRIBUTS PHYSIQUES . LIBELLE/PRESENTATION
: . T LG COL N P RH RV IN PR CO . A
220 : ORUBCI . 003 P .
240 : OECRDE . 003 P .
260 : OECRDR . 003 P .
900 : ZGROUP . A 23 001 Z .
920 : ERMSG . 001 P F .
940 : . A 24 001 L . ENTREE: LISTE, PF1: SUITE,/
960 : . 001 L . PF2: FIN, PF11: AIDE ECRAN,/
980 : . 001 L . PF12: AIDE RUBRIQUE/
O: C2 CH:

```
                                                                        *PTJML.D474.CIV.2020
DESCRIPTION DU SEGMENT : PF00 ZONE COMMUNE DE CONVERSATION

A NLG : CORUB  FORM.INT.  U OCC GR I CMS456 CRNS VALEUR/FCT MAJ/TABLE DOC
  100 : PFSAVE X(115)     D
  120 : PFFRST X          D
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :
      :

*** FIN ***
O: C1 CH: Spf00CE
```

```
                                                                        *PTJML.D474.CIV.2020
ZONES DE TRAVAIL DE L'ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD

DEBUT DU NUMERO DE LIGNE : BA
A NLG S NIVEAU           DESCRIPTION                                        TABLE
  100 * LISTE DES ECRANS NON STANDARD
  110       EXEC PAF DECLARE CU01 CURSOR FOR
  120       SELECT * FROM ECRDEF WHERE
  130       ORUBPL <> I-0010-ORUBP1 OR
  140       ORUBCI <> I-0010-ORUBC1 OR
  150       OECRDE <> I-0010-OECRD1 OR
  160       OECRDR <> I-0010-OECRD2 OR
  170       END-EXEC









O: C1 CH: -W
```

```
--------------------------------------------------------------------------------
|                                                        *PTJML.D474.CIV.2020|
| TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 02   |
|                                                                              |
| A SS NLG OPE OPERANDE                           NVTY CONDITION               |
|          N   INITIALISATIONS PAF                05IT EIBCALEN = 0            |
|      100 EXP INIT                                                            |
|      110 M   '0'     PF00-PFFRST                                             |
| - -- --- --- ------------------------------- ---- -------------------------|
|                                                                              |
|                                                                              |
|                                                                              |
| O: C1 CH: -P02                                                               |
|                                                                              |
--------------------------------------------------------------------------------
```

```
                                                                          *PTJML.D474.CIV.2020
TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 06

A SS NLG OPE OPERANDE                          NVTY CONDITION
         N   CONNEXION PAF                     05IT PF00-PFFRST = '0'
- -- --- --- ------------------------------- ---- --------------------------
  BA     N   MODIF. INDICATEUR PREM. PASSAGE  10BL
  BA 100 M   '1'   PF00-PFFRST
- -- --- --- ------------------------------- ---- --------------------------
  CA     N   CONNEXION DU CURSEUR             10BL
  CA 100 EXP CONNECT CU01 TO
  CA 110     USER    = 'USER'
  CA 120     PASS    = 'PASS'
  CA 130     LIB     = 'CIV'
  CA 140     SESSION = SPACES
  CA 150     NET     = I-0010-PAFVUE
  CA 160     SIZE    = 10
  CA 170     BASE    = 'D474'
  CA 180     IDENT   = EIBTRMID
- -- --- --- ------------------------------- ---- --------------------------
  DA     N   OUVERTURE DU CURSEUR             10BL
  DA 100 EXP OPEN CU01

O: C1 CH: -P06
```

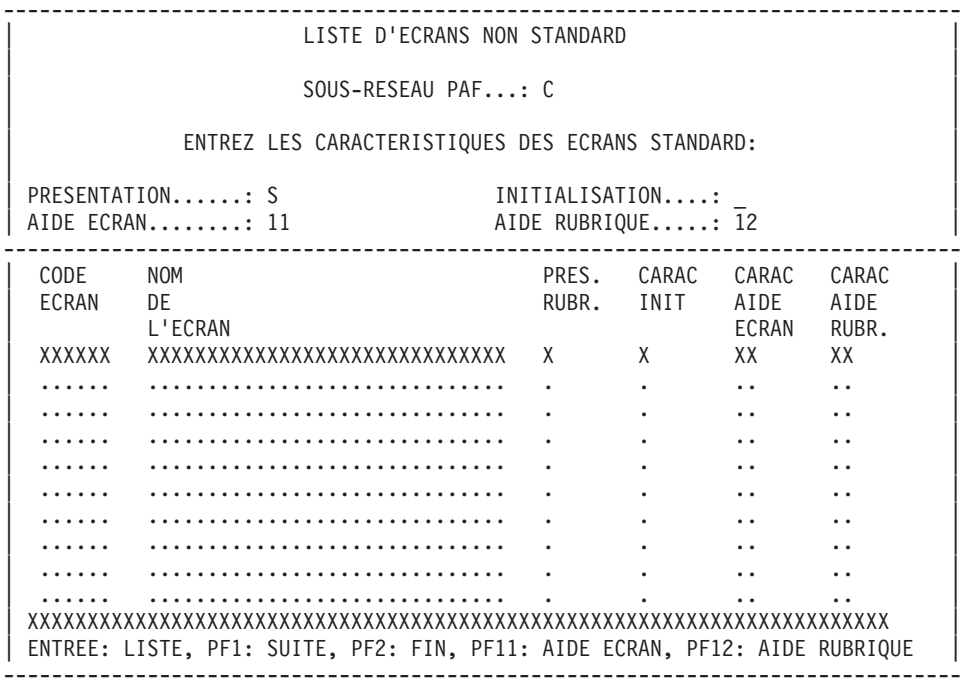
```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 06

A SS NLG OPE OPERANDE                          NVTY CONDITION
  EA     N   SAUVEGARDE CURSEUR EN COMMAREA   10BL
  EA 100 M   CU01-SAVE PF00-PFSAVE
- -- --- --- -------------------------------- ---- -------------------------
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
                                                                              
O: C1 CH: -P06ea
```

```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 40

A SS NLG OPE OPERANDE                          NVTY CONDITION
  31     N   FIN DES TRAITEMENTS PAF           15BL
  31 100 EXP CLOSE CU01
  31 200 EXP QUIT
- -- --- --- -------------------------------- ---- --------------------------
```

```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 52

A SS NLG OPE OPERANDE                          NVTY CONDITION
          N   REINITIALISATION DES ZONES PAF   05BL
      100 M   PF00-PFSAVE CU01-SAVE
- -- --- --- -------------------------------- ---- --------------------------
  BA      N   FERMETURE EN FIN DE CURSEUR      10IT CU01-FT = '1'
  BA 100 EXP CLOSE CU01
- -- --- --- -------------------------------- ---- --------------------------









O: C1 CH: -P52
```

```
                                                                        *PTJML.D474.CIV.2020
 TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 60

 A SS NLG OPE OPERANDE                          NVTY CONDITION
   PF     N   LECTURE DES ECRANS UN A UN        10IT CU01-FT = '0'
   PF 100 EXP FETCH CU01                          AN CATX = 'R'
 - -- --- --- -------------------------------- ---- --------------------------
   PQ     N   AFFICHAGE DU MESSAGE DE FIN       10IT CU01-FT = '1'
   PQ 100 ERU 0001
   PQ 200 GFT
 - -- --- --- -------------------------------- ---- --------------------------









 O: C1 CH: -P60
```

```
                                                                        *PTJML.D474.CIV.2020
TRAITEMENTS ECRAN     O PF0010 LISTE D'ECRANS NON STANDARD    FONCTION: 75

A SS NLG OPE OPERANDE                          NVTY CONDITION
         N   SAUVEGARDE CURSEUR SI PAS D'ERR. 05IT GR-EG = '1'
     100 M   CU01-SAVE PF00-PFSAVE              AN CU01-FT = '0'
- -- --- --- ------------------------------- ---- -------------------------









*** FIN ***
O: C1 CH: -P75
```


LISTE D'ECRANS NON STANDARD
SOUS-RESEAU PAF...: C
ENTREZ LES CARACTERISTIQUES DES ECRANS STANDARD:
PRESENTATION.....: S
INITIALISATION....: _
AIDE ECRAN.....: 11
AIDE RUBRIQUE.....: 12
CODE ECRAN
NOM DE L'ECRAN
PRES. RUBR.
CARAC INIT
CARAC AIDE ECRAN
CARAC AIDE RUBR.
XXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
X
X
XX
XX
ENTREE: LISTE, PF1: SUITE, PF2: FIN, PF11: AIDE ECRAN, PF12: AIDE RUBRIQUE

Chapitre 4. PUF - Pacbase Update Facility

Le mode batch - UPDP / UPGP

La fonctionnalité PUF (Pacbase Update Facility) permet de mettre à jour une Base VA Pac de développement ou d'administration avec des extractions de tables et de colonnes réalisées par le module PAF.

Les procédures batch UPDP et UPGP permettent d'effectuer respectivement la mise à jour batch d'une base de développement et de la base administration à partir d'un fichier séquentiel à l'image des tables PAF.

Pour la description de ces procédures, reportez-vous aux manuels suivants et selon la plate-forme :

- "Les Procédures du Développeur", chapitre "Mises à jour batch", sous-chapitre "UPDP - Mise à jour à partir de tables PAF",
- "Les Procédures de l'Administrateur", chapitre "Gestion de la base Administration", sous-chapitre "UPGP - Mises à jour PAF".

Le mode TP

LE MODE TP

Le mode TP permet de faire des mises à jour en temps réel de flots importants de mouvements sur VisualAge Pacbase à partir d'un programme utilisateur PAF. Sur certaines plates-formes, cela permet de régler les problèmes de mises à jour batch concurrentes avec le TP.

LE SERVICE

Le programme de communication PUF a pour objectif la réception et l'envoi de messages en communication middleware pour satisfaire plusieurs applications.

Suite à une extraction effectuée par PAF, l'application cliente demande la modification d'une ou plusieurs entités, puis lance la mise à jour. Les mouvements concernés sont alors transférés dans le fichier SYSPAF. Ce flot de mouvements peut contenir plusieurs cartes *.

Une fois tous les mouvements transférés, le service PUF est appelé. Il effectue les mises à jour et, si des erreurs sont détectées, il renvoie les mouvements erronés dans le fichier SYSPAF.

L'application cliente demande les types de services suivants :

- Extraction d'entité (download) : le moniteur renvoie la description de l'entité dans la zone de communication.

  Il existe deux sortes de download :

  – download avec intention de mise à jour : la description de l'entité est verrouillée jusqu'à l'upload interdisant ainsi toute autre descente pour mise à jour,

  – download pour consultation : pas de verrou positionné, l'entité reste modifiable.

- Mise à jour d'entité (upload) : le moniteur reçoit une description d'entité qu'il transfère dans le fichier SYSPAF lorsque le dernier message est envoyé ; il lance la transaction de mise à jour et la demande de déverrouillage de l'entité. Ensuite il renvoie les mouvements erronés avec erreurs. Ce service est stocké dans la zone CHOIX, c'est-à- dire la zone de communication ; il comporte 18 caractères et il est codifié de la façon suivante :

  – UPDOWN (1 car.) : D pour download, U pour upload, V pour verrouillage.

  – SLASH (2 car.) : valeur fixe //

  – COMET (1 car.) : code méthode

  – DLIST (2 car.) : LC pour liste par code, LT pour liste par type, LU pour liste utilisateur (uniquement moniteur de communication xxFCOM)

  – DSCODE (3 car.) : code entité locale

  – ENTITE (6 car.) : code de l'entité

  – DSCHXT (2 car.) : sélection de description de l'entité, ** pour toute l'entité, blanc pour la fiche de définition seule

  – UPDWV (1 car.) : cette variable dépend de UPDOWN.

    - en download = W si pas d'intention de mise à jour et X si intention de mise à jour (verrou positionné)

    - en upload = U avec déverrouillage de fin de mise à jour, V sans déverrouillage après mise à jour,

    - en verrouillage = D pour demande de déverrouillage, V pour demande de verrouillage.

- Verrouillage/déverrouillage d'un verrou technique sur certains descriptifs d'entité.

LE POINT D'ENTREE DE PUF-TP

Le programme de suffixe F000 sert de répartiteur pour l'application PUF-TP. Pour le transfert des mouvements, le répartiteur utilise le fichier intermédiaire de suffixe PA, qui est le fichier de travail de PAF ; il envoie les mouvements de mise à jour aux différents programmes dossiers.

Dans le fichier de travail, les enregistrements suivants sont utilisés :

- PA70 qui contient les mouvements envoyés par le programme utilisateur au répartiteur. Ils sont annulés par le répartiteur,
- PA80 qui contient les mouvements erronés (avec la gravité et le libellé de l'erreur), envoyés par le répartiteur au programme utilisateur. Après analyse, ce dernier a à sa charge l'annulation des PA80.

En cas d'ABEND, le répartiteur renvoie un message dans la zone de communication afin de prévenir le programme utilisateur. Le message a la forme suivante :

- ABEND
- code retour sur 2 caractères :

  31 = problème sur le fichier PA

  32 = problème sur les fichiers de VisualAge Pacbase

  99 = problème sur un programme
- nom externe du fichier ou du programme sur 8 caractères.

LE CLIENT PAF

La procédure d'utilisation de PUF-TP est décrite dans le chapitre "Fonctionnement et utilisation dans les Programmes", sous-chapitre "Syntaxe du langage PAF-SQL". Elle consiste à suivre les ordres spécifiques suivants :

- INSERT : stockage des informations
- CALPUF : déclenchement de la mise à jour par appel du répartiteur,

et éventuellement :

- FETCHER : récupération des erreurs signalées par PUF-TP dans le fichier PA80.

LE CLIENT DEPORTE (PUF REMOTE)

Lorsque PUF est serveur, il peut être utilisé à partir d'un client déporté : c'est le PUF-Remote.

Il est possible d'utiliser les communications middleware. L'utilisateur doit créer son propre moniteur de communication dont l'objet sera de réceptionner le message venant du client et de transmettre les informations au répartiteur. Ces informations doivent être stockées dans le fichier de travail PAF.

Le moniteur de dialogue a une zone de communication de la même structure que celle des écrans générés par Pacbench C/S. Il est activé par l'application cliente.

Liste des ordres et fonctionnement

ORDRES ″PUF″ INDEPENDANTS DES CURSEURS (TP SEULEMENT)

Pour l'utilisation de PUF, se référer aux ordres spécifiques INSERT, CALPUF et FETCHER ainsi qu'à la clause UPDATE de l'ordre SET, décrits dans le chapitre ″Fonctionnement et Utilisation dans les Programmes″, sous-chapitre ″Syntaxe du Langage PAF-SQL″.

A la rencontre de la clause ″UPDATE = pref″ les zones suivantes sont générées :

```
01               pref-PUFCOM.
  05             pref-PUFID   PIC X(25) VALUE SPACE.
  05             pref-PUFRET.
    10           pref-PUFPB   PIC X(05) VALUE SPACE.
    10           pref-PUFRC   PIC X(02) VALUE SPACE.
    10           pref-PUFNX   PIC X(08) VALUE SPACE.
    10           pref-FILLER  PIC X(15) VALUE SPACE.
  05             pref-PUFTRA  PIC X(04) VALUE SPACE.
  05             pref-FILLER  PIC X(11) VALUE SPACE.
```

où

PUFID : est l'identifiant de la conversation TP utilisant PAF

(similaire au paramètre IDENT de l'ordre CONNECT).

PUFRET : contient les zones retournées par PUF :

PUFPB : un problème a été détecté (ABEND ou ERROR).

PUFRC : code retour.

PUFNX : nom externe du fichier incriminé.

FILLER : de 15 caractères inutilisés pour le moment.

PUFTRA : est le code transaction de la base VA Pac

(similaire au paramètre BASE de l'ordre CONNECT)

```
01               pref-PAFCOM.
  05             pref-PAFID   PIC X(25)  VALUE SPACE.
  05             pref-PAFTRA  PIC X(04)  VALUE SPACE.
  05             pref-FILLER  PIC X(16)  VALUE SPACE.
  05             pref-PAFRC   PIC X(02)  VALUE SPACE.
  05             pref-ORDER   PIC X(01)  VALUE SPACE.
  05             pref-FI      PIC X(01)  VALUE ZERO.
  05             pref-FT      PIC X(01)  VALUE ZERO.
  05             pref-PUFERR.
```

```
   10            pref-PUFENR.
     15          pref-PUFMVT  PIC X(01)  VALUE SPACE.
     15          pref-PAFTAB  PIC X(10)  VALUE SPACE.
     15          pref-PAFEXT  PIC X(1055) VALUE SPACE.
   10            pref-PUFGRE  PIC X(01)  VALUE SPACE.
   10            pref-PUFLIE  PIC X(66)  VALUE SPACE.
 05              pref-NUMERR  PIC 9(06)  VALUE ZERO.
 05              pref-NUMENR  PIC 9(06)  VALUE ZERO.
 05              pref-FILLER  PIC X(11)  VALUE SPACE.
```

où

PAFID : est l'identifiant de la conversation TP utilisant PAF (similaire au paramètre IDENT de l'ordre CONNECT)

PAFTRA : est le code transaction de la base VA Pac (similaire au paramètre BASE de l'ordre CONNECT)

FILLER : de 16 caractères inutilisés

PAFRC : est le code retour de l'extracteur PAF (70 sur INSERT : l'enregistrement existe déjà)

ORDER : est le code ordre pour l'extracteur PAF (″I″ pour INSERT; ″E″ pour FETCHER)

FI : est la fin de lecture des erreurs: (1= lecture du dernier enregistrement;0= sinon)

FT : est la fin de traitement : (1= lecture au delà du dernier enreg.;0= sinon)

Et : suivant l'ordre donné à l'extracteur PAF ...

PUFERR : contiendra l'enregistrement lu grâce à l'ordre FETCHER. Soit les 5 zones suivantes :

PUFMVT : code du mouvement erroné

PAFTAB : code table PAF du mouvement erroné

PAFEXT : mouvement erroné

PUFGRE : gravité de l'erreur

PUFLIE : libellé de l'erreur

Ou bien ...

PUFENR : contiendra l'enregistrement à écrire grâce à l'ordre INSERT. Soit les 3 zones suivantes :

PUFMVT : code du mouvement ('M', 'C' ,'A', 'B' ou 'S')

PAFTAB : code table PAF du mouvement

PAFEXT : mouvement constitué de 9 caractères techniques suivi de la partie spécifique (image de la description de la table PAF avec toutes les zones renseignées)

Dans tous les cas,

NUMERR : est le numéro d'ordre de l'enregistrement ″erreur″

NUMENR : est le numéro d'ordre de l'enregistrement à écrire ; cette zone est à alimenter avant l'ordre INSERT et elle est à la charge de l'utilisateur.

FILLER : de 11 caractères inutilisés.

Chapitre 5. Mise en oeuvre de PAF par matériel

Introduction

Le principe de PAF implique la transformation des requêtes SQL d'accès à la base VA Pacbase, écrites dans les programmes utilisateur, par la génération de données et d'appels de sous-programmes d'accès à la base dans le source Cobol généré de ces programmes.

Le pré-processeur (BVPAFP10) traite les programmes générés pour effectuer cette transformation.

L'utilisateur dispose de plusieurs manières pour traiter ses programmes générés utilisant PAF :

- utiliser la procédure GPRP qui enchaîne directement le pré-processeur sur l'ensemble du flot généré
- utiliser la procédure PPAF :
- par l'appel de cette procédure dans les commandes de contrôle optionnelles avant/après programme, combinées au JCL de compilation-link-edit,
- par l'appel de cette procédure derrière l'exécution de la procédure GPRT standard d'où sera repris le flot généré,
- par toute autre méthode la plus adaptée aux contraintes de l'exploitation sur le site.

(Voir la section consacrée à PPAF au chapitre "Editions-Générations", sous-chapitre "GPRT : Editions/Générations" du manuel "Les Procédures du Développeur".)

Des sous-programmes PAF pour le batch et le TP sont fournis à l'installation.

LE DICTIONNAIRE PAF

Dans le modèle tabulaire PAF, les données sont organisées dans des tables, les tables sont structurées en colonnes.

Nous tenons à votre disposition un dictionnaire PAF qui contient une représentation des tables PAF sous forme d'entités Pacbase, à savoir, des Rubriques, des Segments et des Structures de Données, un Segment représentant une table, une Rubrique représentant une colonne.

Vous pouvez télécharger ce dictionnaire, en français ou anglais, depuis le site internet de notre Support,

www-306.ibm.com/software/awdtools/vapacbase/support.html

Dans la catégorie 'VisualAge Pacbase downloads', cliquez sur 'Downloads VisualAge Pacbase en français' ou 'VisualAge Pacbase downloads in english'.

Dans le tableau 'Download packages', cliquez dans la colonne 'Download options' de la ligne 'Dictionnaire PAF'.

Le dictionnaire PAF est fourni sous forme de mouvements batch. L'introduction de ce dictionnaire dans la base VisualAge Pacbase est effectuée via la procédure de mise à jour batch UPDT.

Il est fortement conseillé de créer un réseau indépendant de Bibliothèques afin d'éviter les problèmes de compatibilité du dictionnaire du site avec les entités du dictionnaire PAF.

NOTE : Le dictionnaire PAF ne vous est pas nécessaire pour faire une extraction avec le module PAF.

Il vous permet, avec ses descriptions sous forme de Segments, de travailler sur les résultats d'extraction dans un autre programme.

Exemple : vous souhaitez extraire tous les programmes pour modifier le code des cartes avant.

PGM1 : extracteur PAF Select ... PGMDEF, écriture du résultat dans un fichier.

PGM2 : Lecture du fichier résultat, modification de la zone ″Carte avant″ et formattage entrée UPDP.

Dans ce cas la structure du fichier intermédiaire sera PG01.

Version OS/390-CICS

Particularités du système

En TP, il est recommandé d'utiliser la zone EIBTRMID pour identifier les utilisateurs dans les ordres CONNECT (paramètre IDENT).

De plus, il est possible d'accéder à plusieurs bases depuis le même programme PAF, dans le cas où plusieurs bases VA Pac coexistent dans le

même CICS. Le code d'appel de la base (4 caractères) alimentant le paramètre BASE de l'ordre CONNECT permet de sélectionner la base que l'on désire interroger.

Ainsi, un ordre CONNECT typique sous CICS sera :

```
EXEC PAF CONNECT C001 TO
     USER    = ZC00-CUSR
     PASS    = ZC00-CPSW
     LIB     = ZC00-CLIB
     SESSION = ZC00-CSES
     NET     = ZC00-TVIS
     SIZE    = ZC00-LSCR
     IDENT   = EIBTRMID
     BASE    = ZC00-CBAS
```

Le fichier de travail nécessaire au fonctionnement de PAF TP a un DDNAME imposé sous CICS de la forme BVPPA. Il doit être unique pour l'ensemble des programmes accédant aux bases installées sur le site.

Exemple d'exécution d'un extracteur PAF

```
000100 //*************************************************************
000120 //* VISUALAGE PACBASE
000130 //*
001100 //*             - JCL EXAMPLE  -
001200 //*    EXECUTION OF A USER P.A.F. BATCH PROGRAM
001300 //*************************************************************
001400 //PAFBATCH PROC BASE=$BASE,  CODE OF VAPAC DATABASE
001600 //      INDSV='$INDSV',      INDEX OF SYSTEM VSAM FILES
001700 //      INDSN='$INDSN',      INDEX OF SYSTEM NON VSAM FILES
001800 //      INDUV='$INDUV',      INDEX OF USER VSAM FILES
001900 //*:    VSAMCAT='$VCAT',     USER VSAM CATALOG
002000 //*:    SYSTCAT='$SCAT',     SYSTEM VSAM CATALOG
002100 //      STEPLIB=,            USER LIBRARY OF LOAD-MODULES
002200 //      OUT=$OUT             OUTPUT CLASS
003000 //*************************************************************
003100 //MAXKEY  EXEC PGM=IDCAMS
003200 //***********************
003300 //*:STEPCAT DD DSN=&VSAMCAT,DISP=SHR
003400 //SYSPRINT DD SYSOUT=&OUT
003500 //SYSPAF   DD DSN=&&SYSPAF,DISP=(NEW,KEEP),
003600 //         SPACE=(CYL,(3,3)),
003700 //         LRECL=1031,RECORG=KS,KEYOFF=0,KEYLEN=12
003800 //MAXKEY   DD DSN=&INDSN..BVPSY(MAXKEY),DISP=SHR
003900 //SYSIN    DD DSN=&INDSN..BVPSY(REPRO999),DISP=SHR
005000 //WITHPAF  EXEC PGM=------
005100 //***********************
005200 //STEPLIB  DD DSN=&STEPLIB,DISP=SHR
005202 //         DD DSN=$BCOB,DISP=SHR
005300 //*:STEPCAT  DD DSN=&VSAMCAT,DISP=SHR
005400 //PAC7AN   DD DSN=&INDUV..&BASE.AN,DISP=SHR
005500 //PAC7AR   DD DSN=&INDUV..&BASE.AR,DISP=SHR
005600 //PAC7AE   DD DSN=&INDSV..BVPAE,DISP=SHR
005610 //PACGGN   DD DSN=&INDSV..BVPGN,DISP=SHR
```

```
005620 //PACGGR   DD DSN=&INDSV..BVPGR,DISP=SHR
005630 //PACGGU   DD DSN=&INDSV..BVPGU,DISP=SHR
005800 //SYSPAF   DD DSN=&&SYSPAF,DISP=(OLD,KEEP)
005900 //------   DD DSN=---
006000 //------   DD DSN=---
006100 //------   DD DSN=---
006200 //SYSOUT   DD SYSOUT=&OUT
006300 //SYSUDUMP DD SYSOUT=&OUT
007000 //         PEND
008000 //PAFBATCH EXEC PAFBATCH
```

Version IMS/VS/ESA

Particularités du système

LES SOUS-PROGRAMMES D'EXTRACTION

Tout programme PAF IMS (batch ou TP) doit toujours appeler l'extracteur TP. (La version IMS ne comporte pas d'extracteur batch ; c'est l'extracteur TP qui est utilisé en batch et en TP.)

- 'BVPTPST' --> extracteur toutes entités sauf mots-clés.
- 'BVPTPWS' --> extracteur entités mots-clés seulement

Vous devez coder systématiquement l'ordre 'SET' dans la WORKING de vos programmes batch (et TP si ceux-ci sont développés à l'aide du Langage Batch).

Cet ordre doit-être défini de la manière suivante :

```
EXEC PAF SET TYPE = 01
END-EXEC
```

D'autre part, vous devez définir les PCB des bases suivantes :

- éventuellement, les PCB des bases utilisateur,
- OBLIGATOIREMENT, les PCB des bases du système VA Pac :

```
. AN AR AE LB AY AJ PA et DC
.  GU GN GR GY GJ
.  TR WS SV
```

Ces PCBs doivent être définis dans le PSB de la façon suivante :

```
                          .
                          .
                          .
PCB     TYPE=DB,DBDNAME=BDAN$BASE,PROCOPT=GOT,KEYLEN=49
SENSEG  NAME=PAC7AN
PCB     TYPE=DB,DBDNAME=BDAR$BASE,PROCOPT=GOT,KEYLEN=08
SENSEG  NAME=PAC7AR
PCB     TYPE=DB,DBDNAME=BVPDAE,PROCOPT=GOT,KEYLEN=12
SENSEG  NAME=PAC7AE
```

```
PCB     TYPE=DB,DBDNAME=BVPDLB,PROCOPT=GOT,KEYLEN=23
SENSEG  NAME=PAC7LB
PCB     TYPE=DB,DBDNAME=BDAY$BASE,PROCOPT=GOT,KEYLEN=08
SENSEG  NAME=PAC7AY
PCB     TYPE=DB,DBDNAME=BDAJ$BASE,PROCOPT=GOT,KEYLEN=08
SENSEG  NAME=PAC7AJ
PCB     TYPE=DB,DBDNAME=BDPA$BASE,PROCOPT=A,KEYLEN=37
SENSEG  NAME=PAC7PA
PCB     TYPE=DB,DBDNAME=BDDC$BASE,PROCOPT=GOT,KEYLEN=31
SENSEG  NAME=PAC7DC
PCB     TYPE=DB,DBDNAME=BVPDGU,PROCOPT=GOT,KEYLEN=8
SENSEG  NAME=PACGGU
PCB     TYPE=DB,DBDNAME=BVPDGN,PROCOPT=GOT,KEYLEN=49
SENSEG  NAME=PACGGN
PCB     TYPE=DB,DBDNAME=BVPDGR,PROCOPT=GOT,KEYLEN=08
SENSEG  NAME=PACGGR
PCB     TYPE=DB,DBDNAME=BVPDGY,PROCOPT=GOT,KEYLEN=8
SENSEG NAME=PACGGY
PCB    TYPE=DB,DBDNAME=BVPDGJ,PROCOPT=GOT,KEYLEN=08
SENSEG NAME=PACGGJ
PCB    TYPE=DB,DBDNAME=BVPDTR,PROCOPT=GOT,KEYLEN=08
SENSEG NAME=PAC7TR
PCB    TYPE=DB,DBDNAME=BVPDWS,PROCOPT=GOT,KEYLEN=37
SENSEG NAME=PAC7WS
PCB    TYPE=DB,DBDNAME=BVPDSV,PROCOPT=GOT,KEYLEN=15
SENSEG NAME=PAC7SV
PSBGEN  PSBNAME=nompsb,LANG=COBOL
END
```

Avec $BASE = code de la Base VA Pac choisi lors de l'implantation du système VisualAge Pacbase.

IMPORTANT : Les ordres d'appel à l'extracteur (générés par le préprocesseur) sont de la forme :

CALL 'extracteur' USING S-PCB-AN S-PCB-AR S-PCB-AE S-PCB-LB S-PCB-AY S-PCB-AJ S-PCB-PA

S-PCB-DC S-PCB-GU S-PCB-GN S-PCB-GR S-PCB-GY S-PCB-GJ

S-PCB-TR S-PCB-WS S-PCB-SV

nom-du-curseur.

En conséquence, il faut donner les mêmes noms aux PCB dans la 'LINKAGE SECTION' et la 'PROCEDURE DIVISION USING' des programmes batch (S-PCB-xx). Pour les écrans dialogue, les fiches des PCBs appelés dans le PSB du dialogue doivent être nommées PCB-xx (le générateur dialogue ajoutera le préfixe 'S-' dans le source généré au niveau de la 'LINKAGE SECTION' et de la 'PROCEDURE DIVISION USING').

EXTRACTION SOUS LE CONTROLE D'UN SYSTEME DE SECURITE

Les extractions d'entités peuvent s'effectuer sous le contrôle d'un système de sécurité (ex : RACF). Dans ce cas, l'extracteur doit pouvoir différencier le type (batch ou TP) du programme demandant les extractions. En effet, la méthode de contrôle du code utilisateur est différente en batch et en TP.

En batch, le code utilisateur (donné dans l'ordre CONNECT) est contrôlé directement par rapport au système de sécurité par l'intermédiaire d'un sous-programme assembleur (PACSECB) interne à l'extracteur (transparent pour l'utilisateur).

En TP, le code utilisateur est contrôlé par rapport à celui donné dans l'IO/PCB par le système de sécurité.

Pour réaliser ce contrôle, l'extracteur doit nécessairement connaître le type du programme qui l'appelle (batch ou TP). Pour ce faire, le paramètre MODE doit être codé dans l'ordre SET :

- MODE = 'TP ' <- le pgm utilisateur est un pgm TP.
- MODE = 'BATCH' <- le pgm utilisateur est un pgm Batch.

IMPORTANT : Les ordres d'appel à l'extracteur (générés par préprocesseur) dans le cas d'un programme TP sont alors générés de la façon suivante :

CALL 'extracteur' USING S-PCB-AN S-PCB-AR S-PCB-AE S-PCB-LB S-PCB-AY S-PCB-AJ S-PCB-PA

S-PCB-DC S-PCB-GU S-PCB-GN S-PCB-GR S-PCB-GY S-PCB-GJ

S-PCB-TR S-PCB-WS S-PCB-SV

nom-du-curseur S-IPCB.

Pour le batch, les ordres d'appel sont générés de la même façon avec ou sans le contrôle d'un système de sécurité (pas de paramètre S-IPCB dans le CALL).

Exemple d'exécution d'un extracteur PAF

```
000100 //*************************************************************
000120 //* VISUALAGE PACBASE
000130 //*
001100 //* INSTALLATION :    - JCL EXAMPLE  -
001200 //*    EXECUTION OF A USER P.A.F. BATCH PROGRAM
001300 //*************************************************************
001400 //PAFBATCH PROC BASE=$BASE,  CODE OF VAPAC DATABASE
001600 //         INDSV='$INDSV',      INDEX OF SYSTEM VSAM FILES
001700 //         INDSN='$INDSN',      INDEX OF SYSTEM NON VSAM FILES
001800 //         INDUV='$INDUV',      INDEX OF USER VSAM FILES
```

```
001900 //*:       VSAMCAT='$VCAT',     USER VSAM CATALOG
002000 //*:       SYSTCAT='$SCAT',     SYSTEM VSAM CATALOG
002100 //         STEPLIB=,            USER LIBRARY OF LOAD-MODULES
002200 //         OUT=$OUT,             OUTPUT CLASS
002300 //         PSBLIB='$PSBLIB',    IMS PSBLIB
002400 //         DBDLIB='$DBDLIB',    IMS DBDLIB
002500 //         RESLIB='$RESLIB',    IMS RESLIB
002600 //         PROCLIB='$PRCLIB',   IMS WORKLIB
002700 //        BUF=40,SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,SRCH=0,
002800 //        CKPTID=,MON=N,LOGA=0,FMTO=T,DBRC=$DBRC,IRLM=$IRLM
003000 //*************************************************************
003100 //MAXKEY  EXEC PGM=IDCAMS
003200 //***********************
003300 //*:STEPCAT DD DSN=&VSAMCAT,DISP=SHR
003400 //SYSPRINT DD SYSOUT=&OUT
003500 //SYSPAF   DD DSN=&&SYSPAF,DISP=(NEW,KEEP),
003600 //         SPACE=(CYL,(3,3)),
003700 //         LRECL=468,RECORG=KS,KEYOFF=0,KEYLEN=12
003800 //MAXKEY   DD DSN=&INDSN..BVPSY(MAXKEY),DISP=SHR
003900 //SYSIN    DD DSN=&INDSN..BVPSY(REPRO999),DISP=SHR
005000 //WITHPAF  EXEC PGM=DFSRRC00,REGION=$REGSIZ,
005050 //         EXEC PARM=(BMP,------, ... ETC ...
005060 //.....
005100 //.....
005200 //STEPLIB  DD DSN=&RESLIB,DISP=SHR
005220 //         DD DSN=&STEPLIB,DISP=SHR
005250 //         DD DSN=$BCOB,DISP=SHR
005300 //...
005400 //B7AN$BASE   DD DSN=&INDUV..&BASE.AN,DISP=SHR
005500 //B7AR$BASE   DD DSN=&INDUV..&BASE.AR,DISP=SHR
005600 //BVP7AE   DD DSN=&INDSV..BVPAE,DISP=SHR
005610 //BVP7GN   DD DSN=&INDSV..BVPGN,DISP=SHR
005620 //BVP7GR   DD DSN=&INDSV..BVPGR,DISP=SHR
005630 //BVP7GU   DD DSN=&INDSV..BVPGU,DISP=SHR
005800 //B7PA$BASE   DD DSN=&INDUV..&BASE.PA,DISP=SHR
005900 //B7P1$BASE   DD DSN=&INDUV..&BASE.P1,DISP=SHR
006000 //------   DD DSN=---
006100 //------   DD DSN=---
006200 //*
006300 //
```

Version Windows/NT

Particularités du système

Le pré-processeur est constitué d'un programme,BVPAFP10.EXE, installé dans le répertoire des programmes du serveur VA Pac, "\SYS\PGM".

LES SOUS-PROGRAMMES D'EXTRACTION

Les Programmes Utilisateur PAF batch ou TP, générés avec une variante 3 (adaptation au COBOL Micro Focus), utilisent les mêmes sous-programmes d'extraction.

Il y a trois sous-programmes d'extraction :

- BVPBBTST (pour les extractions standard) et BVPBBTWS (pour les extractions de mots clés) sont appelés dynamiquement par les programmes PAF utilisateur.
- BVPBFILE est appelé dynamiquement par les extracteurs (BVPBBTST ou BVPBBTWS) pour accéder à la base VA Pac et au fichier de travail PAF.

Ces trois sous-programmes, appelés dynamiquement, sont livrés compilés et linkés (fichiers .DLL) et sous forme de source COBOL (fichiers .CBL). Les fichiers .DLL sont installés dans le répertoire des programmes , ″\SYS\PGM″, des serveurs VA Pac.

Exemple de compilation : CBLLINK -D [nom_programme].CBL

Les fichiers source COBOL des extracteurs sont installés sous le répertoire assigné dans la procédure.

Les extracteurs appelés dynamiquement doivent être compilés et linkés sur le site lorsque la version du compilateur Micro Focus du site est incompatible avec celle utilisée pour la version VA Pac.

La liste suivante indique les directives de compilation qui doivent être utilisées pour compiler les sous-programmes extracteurs (Micro Focus Net Express V3) :

```
*          - VERBOSE
*          - NOANIM
*          - ASSIGN"EXTERNAL"
*          - NOBOUND
*          - IBMCOMPDIR
*          - NOLIST
*          - NATIVE"EBCDIC"
*          - VSC2E
*          - PERFORM-TYPE"OSVS"
*          - SEQUENTIAL"LINE"
*          - LINKCOUNT"96"
*          - NOTRACE
*          - NOWARNING
*          - CALLFH"FHREDIR"
*          - NOTRUNC
*          - NOCOBOLDIR
*          - NOBELL
*          - INITCALL"UTILS"
```

EXEMPLE DE COMPILATION ET LINK DE PROGRAMMES PAF

Le but est de compiler un programme batch.

Le compilateur utilisé pour les programmes VA Pac est Micro Focus Net Express V3. Il est conseillé d'utiliser le même niveau de compilateur Micro Focus pour les programmes PAF afin d'éviter les conflits entre bibliothèques Micro Focus.

Directives de compilation :

```
ASSIGN "EXTERNAL"
SEQUENTIAL "LINE"
```

Commande de compilation et link :

```
CBLLINK -E [nom_programme].CBL
```

SCENARIO DE PROCEDURE D'EXECUTION APPELANT PAF

On souhaite exécuter un programme batch.

```
*** Destruction du fichier PAF précédent ******
*** Assignation des fichiers de la base  et PAF ***
*** Assignation des fichiers utilisateur ***
 (ajouter les fichiers spécifiques du pgm utilisateur)
*** Exécution du programme
*** Fin
*** Gestion Erreur sur exécution
```

Exemple d'exécution d'un extracteur PAF

```
000100 REM * -------------------------------------------------------
000200 REM *      VISUALAGE PACBASE
000300 REM *
000400 REM * -------------------------------------------------------
000500 REM *         - EXAMPLE OF PAF PROCEDURE EXECUTION -
000600 REM *
000700 REM * -------------------------------------------------------
000800
000900 Set WshShell = Wscript.CreateObject("Wscript.Shell")
001000 Set Args = Wscript.Arguments
001100 Set WshEnv = WshShell.Environment("PROCESS")
001200 Set WshVolEnv = WshShell.Environment("VOLATILE")
001300
001400 Dim FSO
001500 Set  FSO = CreateObject("Scripting.FileSystemObject")
001600 Dim ObjProcess
001700 Set  ObjProcess = CreateObject("BvpWsh.Process")
001800
001900 Rep_BVP="HKLM\SOFTWARE\IBM\BVP VisualAge Pacbase\Server\BVP"
002000
```

```
002100 Rep_Proc = WshShell.RegRead(Rep_BVP & "_SYS\PROC\")
002200 Rep_NLS = WshShell.RegRead (Rep_BVP & "_SYS\NLS\")
002300 Rep_SKEL = WshShell.RegRead (Rep_BVP & "_SYS\SKEL\")
002400 Rep_PGM = WshShell.RegRead (Rep_BVP & "_SYS\PGM\")
002500
002600   WshEnv("PATH") = Rep_PGM & ";" & WshEnv("PATH")
002700
002800 ' Set COBOL environments :
002900 ' ------------------------
003000 Var = "HKLM\SOFTWARE\Micro Focus\NetExpress\"
003100
003200 On Error Resume Next
003300 NetExpress_Version = WshShell.RegRead (Var & _
003400                          "DefaultVersion")
003500 If Err.Number = 0 Then
003600    NetExpress = 1
003700 Else
003800    NetExpress = 0
003900 End if
004000
004100 On Error Resume Next
004200 NetExpress_Version = WshShell.RegRead (Var & "Version")
004300 If Err.Number = 0 Then
004400    ApplicationServer = 1
004500 Else
004600    ApplicationServer = 0
004700 End if
004800
004900
005000 If NetExpress = 1 or ApplicationServer = 1 Then
005100    Var = Var & NetExpress_Version & "\COBOL\"
005200    COBOL_Version = WshShell.RegRead (Var & "Version")
005300    Var = Var & COBOL_Version & "\Environment\"
005400    COBPATH =  WshShell.RegRead (Var & "PATH")
005500
005600    WshEnv("COBDIR") = WshShell.RegRead (Var & "COBDIR")
005700    WshEnv("COBHNF") = WshShell.RegRead (Var & "COBHNF")
005800    WshEnv("PATH") = COBPATH & ";" & WshEnv("PATH")
005900 Else
006000  wscript.echo "NetExpress or Application Server
006100                              not implemented. "
006200 End if
006300 '============================================================
006400
006500 ' EXECUTION OF PAFPGM
006600 '-----------------------------------
006700
006800 ' 1-Delete PAF File
006900 If  FSO.FileExists(Rep_TMP & "\WPAF.tmp") Then
007000 Set FileD = FSO.GetFile(Rep_TMP & "\WPAF.tmp")
007100 FileD.Delete
007200 End If
007300
007400 ' 2-Assign database files ...
007500 WshEnv("PAC7AE") = Rep_SKEL & "\AE"
```

```
007600 WshEnv("PAC7AR") = Rep_BASE & "\AR"
007700 WshEnv("PAC7AN") = Rep_BASE & "\AN"
007800 WshEnv("PACGGN") = Rep_ABASE & "\AN"
007810 WshEnv("PACGGR") = Rep_ABASE & "\AR"
007820 WshEnv("PACGGU") = Rep_ABASE & "\GU"
007840
007900 ' 2bis- ... and  PAF File
008000 WshEnv("SYSPAF") = Rep_TMP & "\WPAF.tmp"
008100
008200 ' 3-Assign of program  files
008300 ' ......................
008400
008500 ' 4- Execute ...
008600 Return = WshShell.Run("BVPAFPGM.exe ", 1, TRUE)
008700
008800 ' 5- If Error
008900 if err.number <> 0 and Return = 0 then
009000  wscript.echo "Syntax or RunTime Error"
009100 End if
009200
009300 if  Return <> 0 then
009400  wscript.echo "PAFPGM execution error"
009500 End if
009600
009700
009800 Wscript.Quit (Return)
009900
```

Version UNIX

Particularités du système

LES SOUS-PROGRAMMES D'EXTRACTION

Pour les programmes utilisateur générés avec une variante 3 (adaptation au COBOL Micro Focus), les mêmes sous-programmes d'extraction sont utilisés pour les programmes batch et TP. Ces extracteurs sont livrés compilés et linkés (fichiers .int ou .gnt).

Il existe trois sous-programmes d'extraction :

- BVPBBTST (pour les extractions standard) et BVPBBTWS (pour les extractions de mots-clés) sont appelés par les programmes PAF utilisateur.
- BVPBFILE est appelé par les extracteurs (BVPBBTST ou BVPBBTWS) pour accéder à la base VisualAge Pacbase et au fichier de travail PAF.

  Ces sous-programmes doivent être compilés sur le site lorsque la version du compilateur Micro Focus du site est différente de celle utilisée pour VisualAge Pacbase pour UNIX.

  Le niveau du compilateur Micro Focus utilisé pour la version VisualAge Pacbase pour UNIX est le suivant :
- MicroFocus 4.1.10 pour la plateforme TRUE64,

- MicroFocus HP 11.30 pour la plateforme HP9000,
- MicroFocus Objet Cobol 4.1.30 pour la plateforme LINUX
- MicroFocus Objet Cobol 4.1.30 pour la plateforme SUN
- MicroFocus Server Express 2.0.10 pour la plateforme AIX

COMPILATION DES SOUS-PROGRAMMES PAF

La compilation s'effectue à partir des fichiers .int livrés dans le répertoire $PACDIR/system/int.

Exemple de script de compilation des sous-programmes PAF

.!/bin/sh

. Assignation du compilateur cobol :

COBDIR=/usr/lib/cobol

export COBDIR

. Assignation de LIBPATH ou LD_LIBRARY_PATH

LIBPATH=$COBDIR/lib:$LIBPATH

export LIBPATH

. Assignation du PATH :

PATH=$COBDIR/bin:$PATH

export PATH

. Liste des programmes a compiler :

PGM="BVPBBTST.int BVPBBTWS.int BVPBFILE int"

. Lancement de la compilation :

cob -uv $PGM $COBOPT

. Fichiers en sortie: BVPBBTST.gnt BVPBBTWS.gnt BVPBFILE.gnt

Exemple de script de compilation des programmes PAF utilisateurs :

.!/bin/sh

. Assignation du compilateur cobol :

COBDIR=/usr/lib/cobol

export COBDIR

. Assignation de LIBPATH ou LD_LIBRARY_PATH

LIBPATH=$COBDIR/lib:$LIBPATH

export LIBPATH

. Assignation du PATH :

PATH=$COBDIR/bin:$PATH

export PATH

. Directives de compilation :

COBOPT="-C ASSIGN=EXTERNAL -C NATIVE=EBCDIC -C
SEQUENTIAL=LI

COBOPT="$COBOPT -C PERFORM-TYPE=OSVS -C OSVS -C NOBOUND -C
I

COBOPT="$COBOPT -C NESTCALL -C DEFAULTBYTE=32"

. Liste des programmes a compiler :

PGM="PGPAF.cbl PGPAFP.cbl"

. Lancement de la compilation :

cob -uv $PGM $COBOPT

. Fichiers en sortie : PGPAF.gnt et PGPAFP.gnt

EXECUTION D'UN EXTRACTEUR PAF

L'exécution de l'extracteur PAF doit être précédée des assignations de fichiers suivantes :

```
. Fichiers permanents en entrée :
  - Fichier des données VisualAge Pacbase pour UNIX : PAC7AR
  - Fichier des index VisualAge Pacbase pour UNIX   : PAC7AN
  - Fichier des libellés d'erreurs                  : PAC7AE
```

```
. Fichier de travail PAF                          : SYSPAF
. Fichiers utilisateurs s'il y a lieu.
```

Exemple d'exécution d'un extracteur PAF

```
000100 #!/bin/sh
000200 #
000300 # EXEMPLE OF AN EXECUTION PROCEDURE USING A PAF PROGRAM
000400 # ------------------------------------------------------
000500 # (execution of PGPAF program)
000600
000700 # Cobol compiler environment variable :
000800 COBDIR=/usr/local/cobol
000900 export COBDIR
001000
001100 # PATH environment variable :
001200 PATH=.:$COBDIR/bin:$PATH
001300 export PATH
001400
001500 # LIBPATH environment variable :
001600 # (for AIX version)
001700 LIBPATH=/usr/lib:$COBDIR/lib
001800 export LIBPATH
001900
002000 # LD_LIBRARY_PATH environment variable :
002100 # (for HP-UX, OSF1, SunOS or Linux version)
002200 LD_LIBRARY_PATH=/usr/lib:$COBDIR/lib
002300 export LD_LIBRARY_PATH
002400
002500 # VisualAge Pacbase directory environment variable :
002600 PACDIR="/lv00/ll350/paclanx"
002700 export PACDIR
002800
002900 # COBPATH environment variable :
003000 # (current directory + VisualAge Pacbase programs directory)
003100 COBPATH=.:$PACDIR/system/gnt
003200 export COBPATH
003300
003400 # VisualAge Pacbase database name (mandatory) :
003500 BASE=BVAP
003600
003700 # VisualAge Pacbase environment variable (mandatory) :
003800 . $PACDIR/config/$BASE/PAC7AE.ini
003900 . $PACDIR/config/$BASE/PAC7AN.ini
004000 . $PACDIR/config/$BASE/PAC7AR.ini
004100 . $PACDIR/config/$BASE/PACGGN.ini
004200 . $PACDIR/config/$BASE/PACGGR.ini
004300 . $PACDIR/config/$BASE/PACGGU.ini
004400 SYSPAF=./wpaf
004500 export SYSPAF
004600
004700 # User application environment variable :
004800 FILE1=./file1
004900 export FILE1
005000
005100 # Start execution :
```

```
005200 cobrun PGPAF
005300
005400 # Deletion of the temporary files :
005500 if [ -r "$SYSPAF" ]
005600 then
005700    rm $SYSPAF*
005800 fi
```

Chapitre 6. Messages d'erreur

Le translateur

Le translateur peut détecter un certain nombre d'erreurs dans l'écriture des ordres PAF-SQL. Chaque erreur provoque l'édition d'un message d'erreur dans un état de compte-rendu accompagné du numéro de ligne du début de la séquence PAF dans le programme traduit.

La liste de ces messages d'erreur est présentée ci-dessous avec, dans certains cas, une explication complémentaire.

CODE COLONNE INCONNUE : <code-colonne> : Le code <code-colonne> n'est pas une colonne de la table désignée après le FROM (dans la langue choisie).

TROP DE CONDITIONS ELEMENTAIRES DANS LE SELECT : Le nombre maximum (50) de conditions élémentaires a été dépassé dans la requête.

CODE CURSEUR TROP LONG : <code-curseur> : Le code du curseur doit être une chaîne de 4 caractères.

CODE CURSEUR DEJA DECLARE : <code-curseur>

TROP DE CURSEURS DECLARES. Le nombre maximum (100) de curseurs déclarés a été dépassé dans le programme.

CODE CURSEUR INCONNU : <code-curseur>. Un ordre de manipulation de curseur a été rencontré pour un curseur non déclaré dans le programme.

AUCUN ORDRE CONNECT POUR LE CURSEUR : <code-curseur>

AUCUN ORDRE OPEN POUR LE CURSEUR : <code-curseur>

AUCUN ORDRE FETCH POUR LE CURSEUR : <code-curseur>

AUCUN ORDRE CLOSE POUR LE CURSEUR : <code-curseur>

AUCUN ORDRE INIT DANS LE PROGRAMME : <code-curseur>

SEQUENCE TROP LONGUE. Une séquence PAF est un groupe de lignes compris entre les chaînes EXEC PAF (en colonne 12) et END-EXEC. Ce message est envoyé par le translateur lorsque le nombre de lignes dépasse 50.

FIN DE PROGRAMME PENDANT UNE SEQUENCE PAF.

OPERANDE NUMERIQUE A TORT : <opérande>

OPERANDE ALPHANUMERIQUE A TORT : <opérande>

OPERANDE TROP LONG : <opérande>. La longueur maximale d'un opérande est de 120 caractères (pour une constante alphanumérique).

OPERANDE COBOL INCORRECT : <opérande>.

TYPES DES COLONNES DIFFERENTS : <code-col1> <code-col2>. Une condition élémentaire porte sur la comparaison de deux colonnes dont l'une est numérique et l'autre alphanumérique.

PARENTHESE OUVRANTE ABSENTE A TORT. La liste des conditions élémentaires succédant au mot-clé WHERE doit comporter un nombre égal de parenthèses ouvrantes et fermantes. Le nombre total de parenthèses ouvrantes doit être égal au nombre de parenthèses fermantes.

PARENTHESE FERMANTE ABSENTE A TORT. Idem message précédent.

AUCUN ORDRE QUIT DANS LE PROGRAMME.

SYNTAXE INVALIDE : <syntaxe>. La syntaxe précisée dans le message n'appartient pas à la grammaire SQL-PAF.

LITTERAL TROP GRAND : <littéral>. La taille maximum d'un littéral est de 120 caractères.

TROP DE LITTERAUX SUR UNE SEULE LIGNE. Le nombre de littéraux sur une ligne d'une séquence PAF ne doit pas dépasser 40.

FIN DE LITTERAL INCORRECT : <littéral>.

CODE TABLE INCONNUE : <code-table>. Le code <code-table> n'est pas le code d'une table PAF (dans la langue choisie).

CODE TABLE ERRONE, TYPE INCONNU : <type-appel-EU>. Le code d'une table d'EU est incorrect car le code d'appel de l'EU, permettant de construire génériquement le code table, n'existe pas (dans le sous-réseau choisi). Pour plus de détails, voir la description des tables d'EU.

CODE TABLE ERRONE, DESCRIPTION INCONNUE : <Dxx>. Le code d'une table d'EU est incorrect car le numéro de description n'est pas défini au niveau de la Meta-Entité. Pour plus de détails, voir la description des tables d'EU.

PARAMETRES DE CONNEXION A VISUALAGE PACBASE INVALIDES.

DELIMITEUR INCORRECT : <délimiteur>. Le délimiteur précisé dans l'ordre SET doit prendre l'une des deux valeurs : SINGLE ou DOUBLE.

MODE EXECUTION INCORRECT : <mode-exécution>. Le mode d'exécution précisé dans l'ordre SET doit prendre l'une des deux valeurs : BATCH ou TP.

TYPE DE GENERATION INCORRECT : <type-génération>. Le type de génération (code alphanumérique sur 2 caractères) précisé dans l'ordre SET doit être une variante VisualAge Pacbase.

PRESENCE A TORT D'UNE CONDITION SUR COL. : <num-colonne>. Il ne peut y avoir de conditions sur les colonnes 05 (code entité VA Pac), 06 (libellé de l'entité VA Pac) et 07 (mots explicites de l'entité) de la table MOTCLE.

PRESENCE A TORT DE PLUSIEURS CONDITIONS SUR COLONNE 01. Il ne peut y avoir au plus qu'une condition sur la colonne 01 (type d'entité) de la table MOTCLE.

PRESENCE A TORT DE PLUSIEURS CONDITIONS SUR COLONNE 02. Il ne peut y avoir au plus qu'une condition sur la colonne 02 (code appel EU) de la table MOTCLE.

PRESENCE A TORT DE PLUSIEURS CONDITIONS SUR COLONNE 03. Il ne peut y avoir au plus qu'une condition sur la colonne 03 (type mot-clé) de la table MOTCLE.

ABSENCE A TORT D'UNE CONDITION UNIQUE SUR COLONNE 04. Il ne doit y avoir qu'une et qu'une seule condition sur la colonne 04 (critère de recherche WS).

ABSENCE A TORT D'UNE CONDITION SUR COLONNE 01. Dans le cas où il existe une condition élémentaire sur la colonne 02 (code d'appel EU) de la table MOTCLE, il doit obligatoirement exister une condition sur la colonne 01 (type entité).

COMPARATEUR INCORRECT SUR COLONNE : <num-colonne>. Seul le comparateur '=' peut être utilisé dans les conditions élémentaires d'une requête sur la table MOTCLE.

OPERANDE INCORRECT POUR COLONNE : <num-colonne>. L'opérande d'une condition élémentaire d'une requête sur la table MOTCLE ne doit pas être une autre colonne de la table.

UTILISATION DE PAF INTERDITE. Vérifier la clé d'accès.

L'extracteur

Le code retour du sous-programme extracteur est une zone, générée par le translateur, de code <code-curseur>-CODRET.

La valeur 00 du code retour signale qu'aucune erreur n'a été détectée.

Les erreurs possibles sont de trois types :

- Erreurs dans la séquence des ordres PAF,
- Erreurs lors des accès fichiers,
- Erreurs sur les données extraites.

ERREURS DANS LES SEQUENCES DES ORDRES

Codes retour : 01 à 10

Le tableau suivant résume les erreurs possibles dans la séquence des ordres PAF : lorsqu'une séquence est illégale, la case correspondante contient la valeur du code retour. Les ordres en LIGNES précèdent les ordres en COLONNES. "NULL" en ligne signifie qu'aucun ordre n'a été rencontré précédemment.

Par exemple, la première ligne illustre qu'aucun ordre ne peut précéder l'ordre INIT.

	INIT	**CONNECT**	**OPEN**	**FETCH**	**CLOSE**	**QUIT**
NULL		01	01	01	01	01
INIT	02		03	04	05	
CONNECT	02			06	07	
OPEN	02		08			
FETCH	02		08			
CLOSE	02			09	10	
QUIT		01	01	01	01	01

```
01   Initialisations non  effectuées,
02   Initialisations déjà effectuées,
03   OPEN    d'un curseur non connecté,
```

```
04   FETCH   d'un curseur non connecté,
05   CLOSE   d'un curseur non connecté,
06   FETCH   d'un curseur non ouvert,
07   CLOSE   d'un curseur non ouvert,
08   OPEN    d'un curseur non fermé,
09   FETCH   d'un curseur fermé,
10   CLOSE   d'un curseur fermé.
```

ERREURS LORS DES ACCES FICHIERS

Les erreurs détectées lors des accès concernent les fichiers de la base VisualAge Pacbase (index, données ou libellés d'erreurs) ainsi que le fichier intermédiaire de stockage.

Codes retour :

21 : erreur ouverture du fichier des index,

22 : erreur ouverture du fichier des données,

23 : erreur ouverture du fichier des libellés d'erreurs,

24 : erreur ouverture du fichier intermédiaire,

31 : erreur lecture ou écriture du fichier intermédiaire,

32 : erreur lecture d'un fichier VisualAge Pacbase,

40 : erreur lors de la connexion à la base VA Pac,

41 : utilisation de PAF non autorisée (clé d'accès).

45 : code utilisateur inconnu

46 : mot de passe incorrect

47 : code application erroné

48 : session invalide

49 : code utilisateur absent à tort

ERREURS SUR LES DONNEES EXTRAITES

Code retour : 50

Les erreurs sur les données détectées lors de l'extraction sont les erreurs de numéricité des colonnes (donc définies avec un format numérique). Ce type d'erreur peut arriver sur les tables associées aux EU.

En effet, le format interne d'une Rubrique appelée dans la description d'une ME peut être modifié alors que des EU ont déjà été créées pour cette ME. Le contenu de la colonne associée à cette rubrique peut alors être alphanumérique au lieu de numérique.

Dans ce cas, l'enregistrement est considéré valide au sens de la requête et lors du FETCH, il est retourné avec le code erreur 50.

Chapitre 7. Introduction aux fonctionnalités PAF-GDP

Préambule

Le module PAF et l'Extension GDP supportent des fonctionnalités utilisables conjointement. Elles ne se substituent pas aux fonctionnalités existantes de PAF et de GDP, elles les enrichissent et permettent de les faire coopérer. Nous les appelerons Fonctionnalités PAF-GDP.

REMARQUE : Les Fonctionnalités PAF-GDP peuvent également être utilisées indépendamment l'une de l'autre.

Ces Fonctionnalités se décomposent donc en PAF+ et GDP+.

Le lecteur trouvera à la page suivante la liste de tous les manuels et documents qui peuvent être nécessaires à l'utilisation des Fonctionnalités PAF-GDP.

L'utilisation des fonctionnalités PAF-GDP requiert une connaissance approfondie du métamodèle VisualAge Pacbase et (s'ils sont installés) des métamodèles propres aux modules PacDesign ou PacBench de la Station de Travail (liés à la méthode de conception utilisée).

DOCUMENTATION

Vous trouverez ci-dessous la liste exhaustive des manuels et documents qui peuvent être nécessaires à l'utilisation des Fonctionnalités PAF-GDP :

1. Manuel de Référence PAF, suivi en annexe d'un exemple de deux Plans Type d'Extraction, avec comptes rendus de validation édités par la procédure XPAF.
2. La description des tables PAF.

   La Documentation des Tables PAF est disponible à l'adresse suivante :

   www-1.ibm.com/support/docview.wss?rs=37&context=SSEP67&uid=swg27005478.

   Cliquez sur le choix "Module PAF". Des liens vous permettent alors de vous débrancher sur les listes des tables PAF de toutes les entités, regroupées par unité fonctionnelle, c'est-à-dire :

   - les entités du métamodèle classique VA Pac,
   - les entités du modèle eBusiness,
   - les entités gérées dans l'espace administration et les entités de gestion Administrateur liées à une base de développement,

- les entités Méthodes.

Vous trouverez également à cette adresse un index cumulatif des tables de toutes les entités, trié par code table. A partir de cet index, vous pourrez vous débrancher directement sur la description de la table recherchée.

3. Les Manuels des Procédures VisualAge Pacbase.

Objectifs des fonctionnalités PAF-GDP

STRUCTURATION AUTOMATISEE ET MAINTENANCE DOCUMENTAIRE

La raison première de l'existence des Fonctionnalités PAF-GDP est l'apport de fonctionnalités supplémentaires par rapport à l'Extension GDP. Ces fonctionnalités ont été conçues essentiellement pour exploiter les relations existant entre les différentes entités du métamodèle utilisé.

EXEMPLE :

Vous voulez générer les commentaires d'un Ecran, par exemple, les commentaires entrés dans:

- les lignes de Commentaires (-GC) de l'Ecran,
- les lignes de Commentaires (-GC) de ses Segments,
- la Description (-D) des Rubriques appelées par ces Segments.

Avec GDP, vous devez appeler explicitement chaque Segment dans la Description (- D) du Volume.

Lorsqu'un nouveau Segment est appelé dans l'Ecran (-CS), il doit être également ajouté dans la Description du Volume.

Avec PAF-GDP, vous spécifiez l'information qui doit être éditée dans le Volume en définissant -- une seule fois -- un chemin d'extraction, également appelé Plan Type d'Extraction (PTEx).

Dans l'exemple précédent, le chemin d' extraction va commencer par l'entité Ecran, trouver les Segments appelés dans cet Ecran pour aller jusqu'aux Rubriques, son parcours étant guidé par le Métamodèle VA Pac.

PAF-GDP est donc non seulement un outil de constitution automatique de documentation d'applications, mais aussi un outil de maintenance automatisée de cette documentation. Lorsque la composition de l'application documentée évolue, il suffit de re-générer les Documents appropriés sans modifier leur Description.

NORMALISATION DE LA DOCUMENTATION

GDP+ vous permet d'écrire des Plans Type d'Edition (PTEd), c'est-à-dire des squelettes que vous pouvez utiliser dans plusieurs cas :

- Avec GDP, les options d'édition d'un Volume ne sont valables qu'au niveau de ce Volume, ce qui implique qu'elles doivent être spécifiées dans chaque Volume. Ceci ne facilite pas la normalisation d'un ensemble documentaire.

Avec GDP+, vous pouvez indiquer toutes les options d'édition désirées dans un seul Plan Type d'Edition. Il vous suffit ensuite d'appeler le PTEd dans tous les Volumes que vous voulez éditer.

- Avec GDP+, vous pouvez aussi codifier des appels normalisés. Par exemple, suite à l'appel dans un PTEd de

```
TGEN___D__
```

toutes les instances de Texte dont le code débute par les lettres 'GEN' seront éditées dans TOUS les Volumes faisant appel à ce PTEd.

- En outre, et cela est le plus important, vous pouvez utiliser conjointement GDP+ et PAF+ pour normaliser la structure même de la documentation.

C'est dans cette utilisation que l'expression Plan Type d'Extraction prend tout son sens, puisque le PTEd devient un squelette de structuration. Différents Volumes documentant une ou plusieurs applications peuvent être générés à partir du même Plan Type d'Edition (PTEd) qui gère les données extraites par un seul Plan Type d'Extraction (PTEx).

REMARQUE : Un Volume peut utiliser plusieurs Plans Type d'Edition.

CONCLUSION

Avec PAF-GDP, automatisation et normalisation ne sont pas pour autant synonymes de rigidité, puisque vous définissez vous-même les paramètres d'extraction et d'édition. Cependant, cette définition doit émaner d'une seule instance, faute de quoi il n'y aurait pas de normalisation.

Mode opératoire des fonctionnalités PAF-GDP

Les fonctions PAF+/Extraction et GDP+/Edition peuvent être utilisées séparément l'une de l'autre ou conjointement (PAF-GDP).

PAF+ permet l'écriture du Plan type d'extraction et son exploitation quand le PTEx est un Extracteur utilisateur.

GDP+ permet l'écriture et l'exploitation du Plan type d'édition.

Les fonctionnalités PAF-GDP sont mises en oeuvre quand le Plan type d'édition fait appel à un Plan type d'extraction de type macro-commande.

- Lorsque la fonction PAF+/Extraction est utilisée seule, elle permet de générer un Programme extracteur utilisateur avec possibilité de mise en forme des données extraites.
- Lorsque la fonction GDP+/Edition est utilisée seule, elle permet de créer des squelettes de normalisation pour l'édition des rapports (options d'édition standard, textes systématiquement appelés, codification des appels normalisée).
- Lorsque les deux fonctions sont utilisées conjointement, PAF+ extrait des données de la base qui seront ensuite exploitées par GDP+, pour être finalement éditées dans un Volume.

PAF+ : LE PLAN TYPE D'EXTRACTION

La Fonction PAF+ vous permet d'écrire un Plan-Type d'Extraction (PTEx), c'est-à-dire un chemin d'exploration d'une base de développement ou d'administration à partir duquel est généré un moteur. Ce moteur extrait automatiquement les informations désirées.

Pour écrire un Plan Type d'Extraction, vous devez définir et décrire une Entité Utilisateur issue de la Méta-Entité réservée de code .PPTEX, dont le code d'appel est 7E. Le choix d'accès à la Définition d'une telle Entité Utilisateur, par exemple de code MONPTX, sera donc Y7EMONPTX. Dans sa Description (-D), vous spécifierez le chemin d'extraction.

Il existe deux types de PTEx, donc deux types d'Entités Utilisateur issues de .PPTEX :

- Le type E vous permet de générer un Programme Extracteur exécutable de façon autonome.
- Le type M vous permet de générer une macro-commande (sous-programme) qui doit être appelée dans un Plan Type d'Edition (Voir les fonctionnalités GDP+).

VALIDATION

Vous devez ensuite valider le Plan Type d'Extraction par la procédure batch XPAF, qui génère soit le programme Extracteur Utilisateur, soit le sous-programme de la macro-commande.

>>>> La procédure XPAF est documentée dans le manuel "Les Procédures du Développeur", chapitre "Extraction personnalisée / documentation automatisée".

La Validation du PTEx, dans le cas où aucune erreur n'a été signalée, produit un source COBOL que vous devez compiler puis linker afin de le rendre exécutable.

EXECUTION D'UN EXTRACTEUR UTILISATEUR (PTEx DE TYPE E)

Une fois validé, compilé et linké, un Programme Extracteur Utilisateur est prêt à être exécuté.

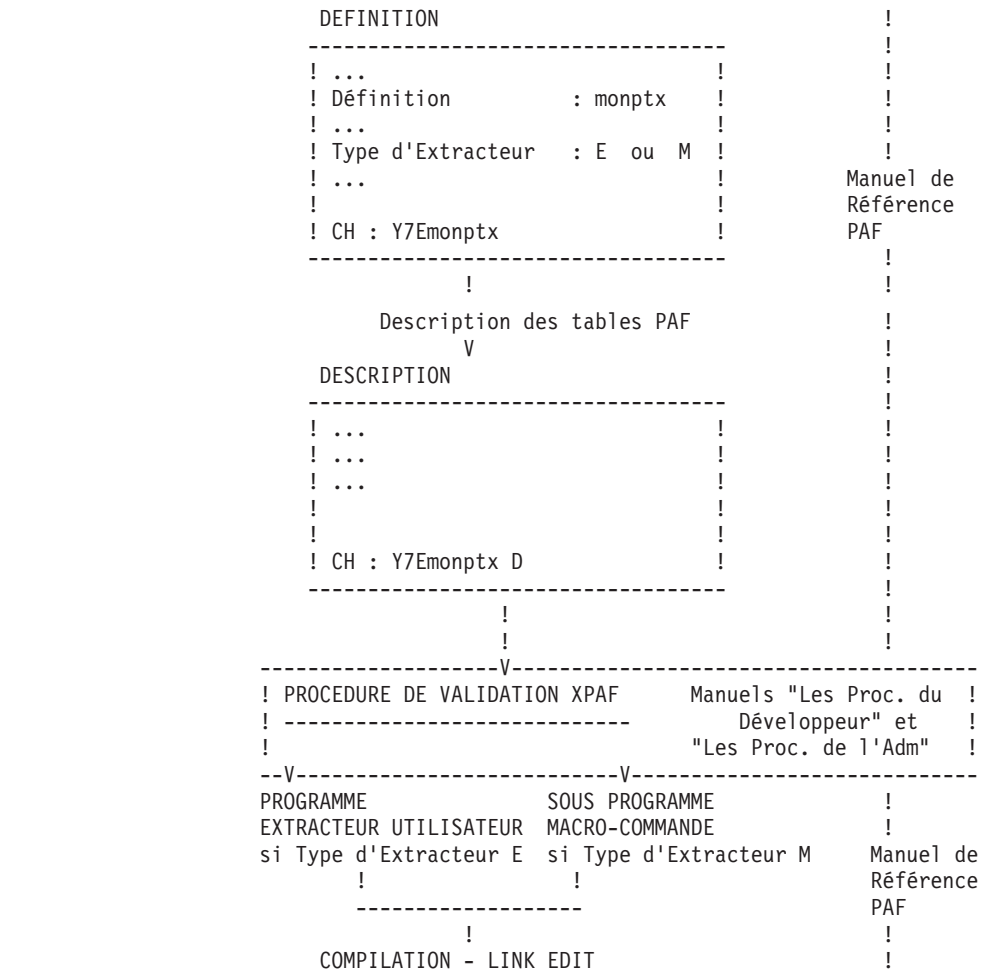
EXECUTION D'UNE MACRO-COMMANDE (PTEx DE TYPE M)

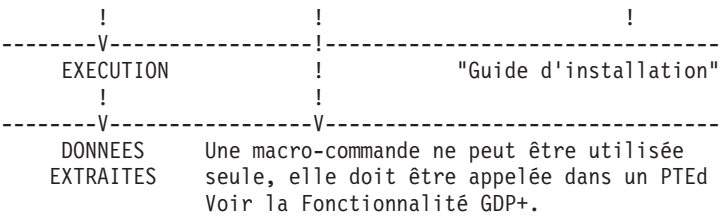
Une fois validée, compilée et linkée, une macro-commande n'est pas prête à être exécutée. Elle doit être appelée dans un Plan type d'édition.

NOTE

Un PTEx est indépendant de la base où il est défini et décrit.

PAF+ : LE PLAN TYPE D'EXTRACTION - SCHEMA D'UTILISATION





GDP+ : LE PLAN TYPE D'EDITION

La Fonction GDP+ permet l'écriture de Plans Type d'Edition supportés par des instances de l'entité Volume de type P.

Un PTEd organise l'édition de données extraites par PAF+ quand sa Description appelle un Plan Type d'Extraction (sur une ligne de type M).

Les Plans Type d'Edition peuvent être réutilisés puisqu'ils peuvent être applelés dans différents Volumes dans lesquels ils peuvent être personnalisés.

****

Une fois défini et décrit, un Plan Type d'Edition doit être validé par la procédure batch XPDM.

>>>> La procédure XPDM est documentée dans le manuel ″Les Procédures du Développeur″, chapitre ″Extraction personnalisée / documentation automatisée″.

****

REMARQUE : Un Plan Type d'Edition ne peut être édité en soi. Il doit être appelé dans un Volume, qui, lorsqu'il est édité, inclut le PTEd.

Cependant, il est possible d'éditer la Description du PTEd (procédure GPRT, commande DCV).

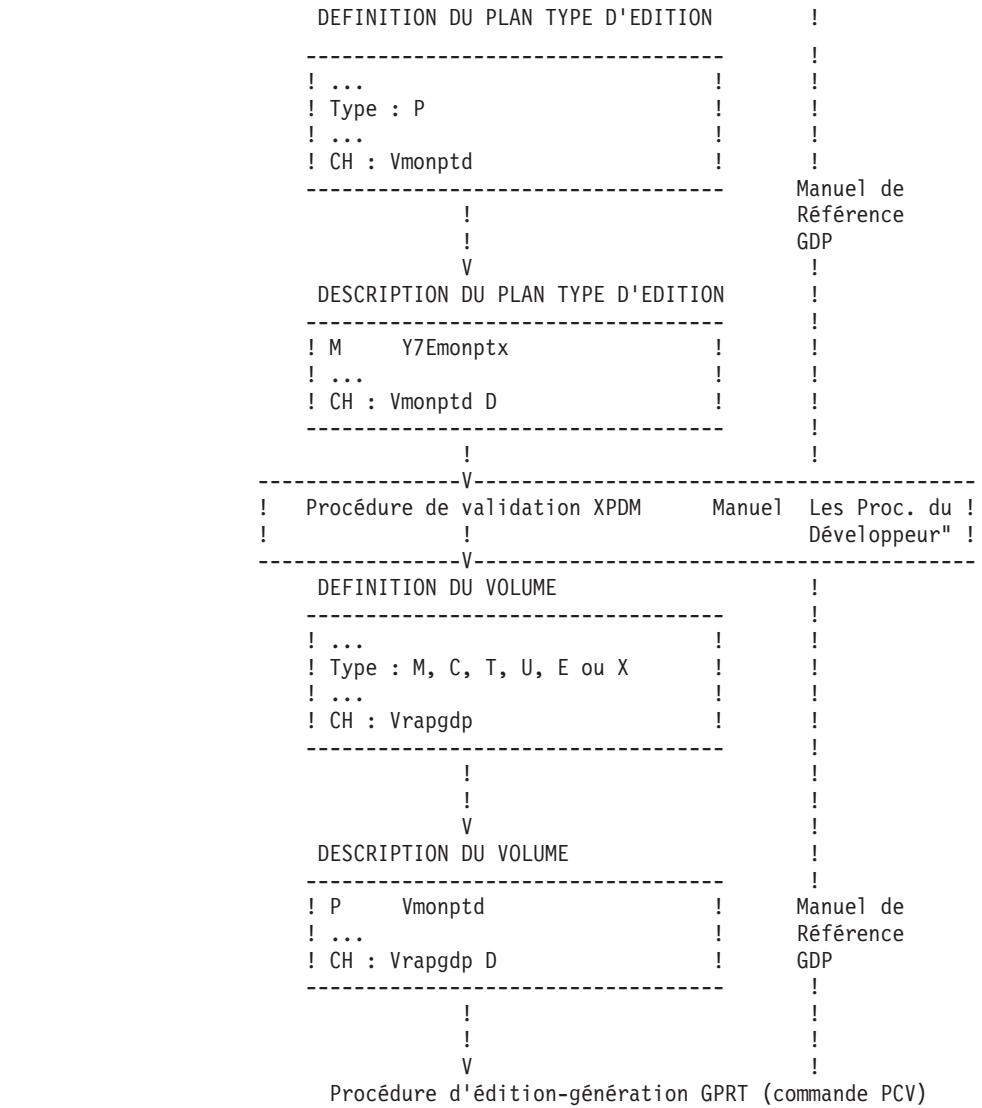
Vous pouvez aussi utiliser la procédure PRGS qui édite le fichier utilisé pour sauvegarder les Plans Type d'Extraction et leurs Plans Type d'Edition

>>>> La procédure PRGS est documentée dans le Manuel ″Les Procédures du Développeur″, chapitre ″Extraction personnalisée / documentation automatisée″.

Un PTEd est indépendant de la Base où il est défini et décrit, tant que le radical est le même. Si un PTEx est modifié et donc re-validé (XPAF), tous les PTEd faisant appel à ce PTEx doivent être éventuellement modifiés et toujours re-validés (XPDM).

GDP+ : LE PLAN TYPE D'EDITION - SCHEMA D'UTILISATION

Ce schéma illustre l'utilisation conjointe de PAF+ et GDP+; le Plan Type d'Edition fait appel à une macro-commande (PTEx de type M).



Chapitre 8. PTEx: Ecrans/zones Définition & Description

Un Plan Type d'Extraction est défini et décrit sur une Entité Utilisateur, instance de la Méta-Entité extension dédiée à PAF+/Extraction. Le code de cette Méta-Entité extension est '.PPTEX' et son type d'appel '7E'.

Le lecteur trouvera dans les pages qui suivent les écrans de Définition (CH: Y7E.....) et Description (CH: Y7E.....D) des Entités Utilisateur dérivant de cette Méta-Entité, ainsi qu'une brève description de leurs zones de saisie.

Il est vivement conseillé de documenter le Plan Type d'Extraction en représentant graphiquement l'arbre d'extraction dans l'écran de commentaires (-GC) de l'Entité Utilisateur et de multiplier les lignes de commentaire dans l'écran de description de l'Entité Utilisateur.

Ces commentaires seront systématiquement édités lors de la soumission de la procédure de validation du Plan Type d'Extraction.

```
Definition                      TYPE : 7E     META-ENTITE :.PPTEX
Code de l'entité u : 1_____
Nom de l'entité ut : 2_____________________________________
Code programme de  : 3_______
Type d'extracteur  : 4
Liste des options  : 5_____________________________________________________
Taille maximale de : 6__
Option de tri      : 7____
```

NU	LO	CLASSE VALEUR	SIGNIFICATION DES RUBRIQUES ET MODE DE REMPLISSAGE
1	6		Code de l'entité utilisateur (ALPHABETIQUE) RUBRIQUE IDENTIFIANT PRESENCE OBLIGATOIRE MAJUSCULE
2	36		Nom de l'entité utilisateur (ALPHABETIQUE) LIBELLE DE L'ENTITE PRESENCE OBLIGATOIRE
3	8		Code programme de l'extracteur (ALPHABETIQUE) PRESENCE OBLIGATOIRE MAJUSCULE
			La valeur renseignée dans cette zone indique où sera stocké le source du Plan Type d'Extraction avant compilation et ce que contient la clause PROGRAM-ID.
			De plus, si le Plan Type d'Extraction est un Programme Extracteur Utilisateur, cette valeur sera le nom externe du programme exécuté.
4	1		Type d'extracteur (ALPHABETIQUE) PRESENCE OBLIGATOIRE MAJUSCULE
		M	Macro-Commande (Sous-Programme)
		E	Extracteur Utilisateur (Programme)
5	50		Liste des options (ALPHABETIQUE)
		STATI	Valeur par défaut: CALL statique de l'extracteur PAF
		DYNAM	CALL dynamique de l'extracteur PAF
6	3		Taille maximale de l'enregistrement (NUMERIQUE)
			Longueur maximum des enregistrements extraits mis en forme. Cette zone ne s'applique donc qu'au Plan Type d'Extraction de type Extracteur Utilisateur (E).
7	5		Option de tri (ALPHABETIQUE) MAJUSCULE
			Deux options sont possibles pour le tri du résultat d'un Extracteur Utilisateur (E):
		CURS	Valeur par défaut: Tri en majeur sur l'occurrence extraite (type et code) et tri en mineur sur le critère d'identification (chemin d'extraction permettant d'arriver à une occurrence).
		IDENT	Tri en majeur sur le critère d'identification et tri en mineur sur l'occurrence extraite (type et code).

NU	LO	CLASSE VALEUR	SIGNIFICATION DES RUBRIQUES ET MODE DE REMPLISSAGE
			NOTE: Pour une Macro-Commande (M), l'option de tri est toujours CURS, quelle que soit la valeur de cette zone.

```
Description                    7E ______                              01
                                  ....................................
Num L T REF. SELECTION                                  VEN.VIEW OPDV
___ _ _ ____ ____________________________________________ ________ ____
1__ 2 3 4___ 5___________________________________________ 6_______ 7___
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
___ _ _ ____ ____________________________________________ ________ ____
```

NU	LO	CLASSE VALEUR	SIGNIFICATION DES RUBRIQUES ET MODE DE REMPLISSAGE
1	3		Numéro de ligne (NUMERIQUE) RUBRIQUE IDENTIFIANT PRESENCE OBLIGATOIRE
2	1		Niveau d'imbrication (ALPHABETIQUE)
			Sa valeur doit être numérique et comprise entre les valeurs 1 et 9 incluses. La valeur 1 est unique dans un "Plan Type d'Extraction" car elle représente le point d'entrée de ce Plan Type.
			Sa présence est autorisée sur les lignes de type S, A, et V.
3	1		Type de ligne (ALPHABETIQUE) PRESENCE OBLIGATOIRE MAJUSCULE
		S	Séquencement
		A	Enchaînement
		I	Conditionnement en entrée de l'extraction
		O	Filtrage en sortie des occurrences extraites
		P	Présentation (ou mise en forme)
		V	Ventilation
		*	Commentaires (imprimés dans le compte-rendu de Validation)
		blanc	Ligne ignorée par la procédure de Validation
			Pour plus de détails, voir le chapitre "PTEx : Le Plan Type d'Extraction" dans ce manuel.
4	4		Code curseur "Référence" (ALPHABETIQUE) MAJUSCULE

NU	LO	CLASSE VALEUR	SIGNIFICATION DES RUBRIQUES ET MODE DE REMPLISSAGE
			Sa présence est obligatoire sur les lignes de type S, A, et V. Il doit être unique. Sur les lignes de type S et A, il spécifie le code curseur associé à une table PAF. Cela permet le séquencement de l'extraction et identifie l'occur- rence extraite lors de la création du critère d'identification. Pour plus de détails, voir le chapitre 'PTEx : Le Plan Type d'Extraction', sous-chapitres ″Séquencement (Lignes 'S') et (Lignes 'A')″.
			Sur les lignes de type V, il définit un nouveau code curseur pour la ventilation d'une occurrence, différent de celui établi dans le chemin d'extraction. Pour plus de détails, voir le chapitre 'PTex : Le Plan Type d'Extraction', sous-chapitre 'Ventilation (lignes 'V')'.
5	40		Sélection ou commentaire (ALPHABETIQUE)
			OBLIGATOIRE pour les lignes de type S, A, P, I et O.
			Pour plus de détails, voir le chapitre ″PTEx : Le Plan type d'Extraction″.
6	8		Curseurs ″Vues″ (ALPHABETIQUE)
			Cette zone regroupe, successivement, les curseurs de ventilation et de mise en forme.
			Pour plus de détails, voir le chapitre ″PTEx : Le Plan Type d'Extraction″, sous-chapitres ″Ventilation (Ligne 'V)' pour le curseur de ventilation et 'Mise en Forme (Ligne 'P')″ pour le curseur de mise en forme.
7	4		Options de traitement (ALPHABETIQUE) MAJUSCULE
			FACULTATIF
			Colonne O : options d'édition (pour les lignes S)
		blanc O	Mémorisation de l'occurrence dans le fichier Résultat d'extraction ; cette donnée est donc éditable.
		N	Occurrence utilisée comme intermédiaire; cette donnée n'est donc pas éditable.
			Colonne P : définition d'un point d'entrée
		P	Définition d'une occurrence comme point d'entrée
			Colonne D : valeur du délimiteur
		blanc	Valeur par défaut dans la syntaxe de description de l'extraction sur les lignes de type S et A
		,	Valeur par défaut sur les lignes de type P Si ces valeurs par défaut ne conviennent pas, l'utilisateur peut les modifier dans cette zone. Attention, les valeurs ″-″ et ″/″ sont interdites.

NU	LO	CLASSE VALEUR	SIGNIFICATION DES RUBRIQUES ET MODE DE REMPLISSAGE
			Colonne V : valeur du délimiteur encadrant une constante sur une ligne de type P (voir chapitre ″PTEx: Le Plan Type d'Extraction″, sous-chapitre ″Mise en Forme (Ligne 'P')″)
		}	Valeur par défaut Si cette valeur ne convient pas, l'utilisateur peut la modifier dans cette zone. Attention, les valeurs ″-″ et ″'″ sont interdites.

Chapitre 9. PTEx: Le Plan Type d'Extraction

Séquencement (lignes 'S')

Les données sont extraites de la Base VA Pac selon un cheminement -- un séquencement -- qui exprime les chaînages existants entre les entités. C'est pourquoi une très bonne connaissance du métamodèle utilisé est requise.

Le métamodèle est soit le métamodèle classique VA Pac, soit le métamodèle implémenté par la Station de Travail VA Pac au travers de ses modules PacDesign ou PacBench et selon la méthode utilisée.

Un PTEx peut donc être envisagé comme un arbre dont les branches explorent la Base de Spécifications à un niveau de plus en plus fin.

La syntaxe utilisée pour décrire le séquencement d'une extraction est proche du langage de navigation en TP.

Autrement dit, les questions à se poser AVANT TOUTE AUTRE CHOSE sont les suivantes:

1. Quelles sont les données à extraire ?
2. Quels sont les choix TP à saisir pour obtenir les écrans correspondants à ces données ?

Des exemples partiels de PTEx sont donnés à la fin de ce sous-chapitre, deux exemples complets sont fournis à la fin de ce manuel.

Le séquencement de l'extraction est saisi sur des lignes de type S dans la zone SELECTION du -D de l'EU.

L'écriture d'un séquencement se compose de trois éléments, le type d'entité, l'occurrence, et le type de ligne.

1. LE TYPE D'ENTITE:

   Le type d'entité à saisir correspond au type d'entité que l'on renseigne en TP dans la zone CHOIX.

   NOTE : Si l'entité est une entité spécifique à la Station de Travail, son type doit être codifié selon le format suivant:

   ```
   //ü__CCC

   où ü est le code Méthode ("M" pour Merise,
   "D" pour YSM) et
   CCC est le code local de l'entité.
   ```

```
Pour connaître la valeur de ces codes,
utilisez la commande PCM saisie dans la zone COM de
l'écran "COMMANDES  D'EDITION ET DE GENERATION"
(CH: GP).

Le code de la méthode sur un caractère sera
saisi dans la zone ENTITE. Les valeurs possibles
pour les Méthodes sont les suivantes :

        M pour Merise

        D pour YSM

        A pour SSADM

        O pour OMT

        F pour IFW
```

1. L'OCCURRENCE :

   L'occurrence doit être séparée du type d'entité par un délimiteur (valeur par défaut : SPACE, modifiable dans la colonne 'D').

   Elle est identifiée grâce à deux éléments séparés par un tiret :

   a. Code curseur identifiant la Table PAF hiérarchiquement supérieure. Le code curseur de référence d'une ligne de niveau hiérarchique ″n″ est systématiquement reporté devant le code occurrence de la ligne de niveau hiérarchique ″n+1″.

      Le cheminement dépend exclusivement du code curseur qui doit être unique dans un Plan Type d'Extraction.

   b. Contenant du code occurrence : code colonne PAF.

      Un index des codes colonnes PAF est disponible sur le site internet du Support,

      www-1.ibm.com/support/docview.wss?rs=37&context=SSEP67&uid=swg27005478 (cliquez sur ″Module PAF″).

      S'il s'agit d'une donnée gérée par la Station de Travail, le code de la colonne est celui de la Rubrique appelée par l'Entité Utilisateur qui supporte cette donnée dans la Base.

      Dans ce cas, on séparera le code curseur et le code Rubrique par deux tirets.

      Pour obtenir les codes des Rubriques composantes des Entités Utilisateur servant à décrire les entités Station, utilisez la commande PCM.

2. TYPE DE LIGNE :

   C'est la simple codification du CHOIX de 2nd niveau en TP (ex : dans //M DOM DOM-COEU X1MCD, le code X1MCD est le type de ligne). Ce code indique à PAF+ la table à extraire.

   Il doit être séparé de l'identifiant de l'occurrence par un délimiteur (valeur par défaut : SPACE, modifiable dans la colonne 'D').

Les codes des tables à extraire des entités gérées par la Station de Travail sont disponibles sur le site internet du Support, à l'adresse suivante:

www-1.ibm.com/support/docview.wss?rs=37&context=SSEP67&uid=swg27005478

EXEMPLES:

```
NLG : L T REF. SELECTION
010 : 1 S PGM  P
020 : 2 S LSDO P PGM-CPGM CD
```

Sur la ligne 010, de niveau d'imbrication '1', l'occurrence n'a pas besoin d'être précisée puisqu'il s'agit du point d'entrée initial.

Sur la ligne 020, pour chaque occurrence de l'entité Programme, on demande l'extraction de la liste de ses Structures de Données.

```
NLG : L T REF. SELECTION
010 : 1 S SEG  S
020 : 2 S LRUB S SEG-CSEG CE
```

On obtient ainsi la liste des Rubriques (curseur LRUB) appartenant à chaque occurrence de l'entité Segment (curseur SEG).

```
NLG : L T REF. SELECTION
010 : 1 S PGM  P
020 : 2 S PGMP P PGM-CPGM P
```

Liste des lignes '-P' de chaque occurrence de l'entité Programmes.

```
NLG : L T REF. SELECTION
010 : 1 S DOM  //M DOM
020 : 2 S LMCD //M DOM DOM-COEU X1MCD
```

Liste des Modèles Conceptuels de Données pour chaque occurrence de l'Entité Domaine.

```
NLG : L T REF. SELECTION
010 : 1 S PHA  //M PHA
020 : 2 S CHA  //M CHA PHA--PHACH
```

Définition du Dessin de Chaîne associé à chaque occurrence de l'entité Phase.

Explication de cette codification:

Sur la Définition d'une Phase figure le code du Dessin de Chaîne associé. Il faut donc extraire la rubrique qui contient le code occurrence du Dessin de Chaîne associé.

Pour la quatrième zone - DESSIN DE CHAINE - on lira le code de la rubrique correspondante dans la colonne CLASSE/VALEUR (.PHACH). Elle sera codifiée -PHACH dans le PTex, séparée du code curseur par un autre tiret.

De façon plus générale, à toute Rubrique qui décrit une Entité Utilisateur ET dont le code est préfixé par un point, devra correspondre un code colonne préfixé par un tiret à la place du point.

NOTE : Vous trouverez en annexe à ce manuel deux exemples complets de Plan Type d'Extraction, l'un utilisant le cheminement propre au métamodèle de la Station de Travail, l'autre exploitant les chaînages propres au métamodèle classique VisualAge Pacbase. Il est intéressant de noter que les résultats obtenus par ces deux PTEx seront identiques.

Séquencement : cas particuliers

Certaines ambiguïtés peuvent apparaître au niveau de la sélection des tables à extraire, en particulier pour les tables de chaînages. Dans ce cas, il est nécessaire d'ajouter l'identifiant d'occurrence et la table.

EXEMPLE :

On veut connaître les utilisations d'un Segment dans les Programmes.

Ces utilisations sont consultées sur l'écran 'S....XP.....CD..' et sur l'écran 'S....XP.....W.....'.

Dans le premier écran, il s'agit des Segments appelés par les Structures de Données appelées par le Programme (P.....CD). Dans le second écran, il s'agit des Segments appelés dans les Zones de Travail du Programme (P.....W).

C'est pour cela que la ligne suivante serait ambiguë :

```
NLG : L T REF. SELECTION
020 : 2 S SEGP S SEG-CSEG XP
```

On précisera donc:

```
              S SEG-CSEG XP PGM-CPGM CD
et/ou         S SEG-CSEG XP PGM-CPGM W
```

Si à ce stade du séquencement, le curseur PGM n'a pas été défini, autrement dit que l'on n'a pas demandé d'extraction sur l'entité Programme, la formulation sera la suivante :

```
              S SEG-CSEG XP * CD
et/ou         S SEG-CSEG XP * W
```

On obtiendra ainsi les utilisations du Segment sur les lignes -CD et/ou -W de tous les Programmes de la bibliothèque interrogée.

Séquencement (lignes 'A')

Une ligne de type 'A', comme une ligne de type 'S', exprime une sélection d'extraction. Pour que l'on puisse conditionner une extraction en testant la valeur d'une Rubrique appartenant au curseur mais n'étant pas identifiante, il faut définir une ligne de type 'A' et renseigner le curseur hiérarchiquement supérieur dans la zone 'VEN.'.

EXEMPLE :

L'objectif est de connaître les utilisations des Rubriques dans les '-CD' des Programmes.

```
NLG : L T REF. SELECTION                          VEN.VIEW
010 : 1 S RUB  E
020 : 2 A PCDE P * CD                             RUB
030 :   O      CRUB = RUB-CRUB
040 :   O      OR CSTE = RUB-CRUB
050 :   S PGM  P PCDE-CPGM
```

Dans cet exemple, les colonnes identifiantes du curseur PCDE sont P (Programme) et CD (Structure de Données); CRUB est une Rubrique supplémentaire de la table.

On extrait d'abord toutes les Rubriques, puis on sélectionne parmi toutes les lignes '-CD' des Programmes celles qui font appel à des Rubriques, les Programmes n'ayant pas fait l'objet d'une extraction préalable.

On voit que le cheminement subit une rupture, l'enchaînement entre Rubriques et Programmes doit donc être assuré par une ligne de type 'A' qui portera dans les quatre premiers caractères de la zone VEN.VIEW le curseur de la Table des Rubriques.

Ce type de ligne comporte donc nécessairement une sélection exprimée à l'aide d'un astérisque.

Une fois que l'on a précisé les objets de l'enchaînement, il reste à expliciter la condition de l'enchaînement. Cette expression, dite de "filtrage", est saisie sur au moins une ligne de type 'O' et doit faire référence au curseur spécifié sur la ligne de type 'A' (zone VEN.VIEW).

NOTE : Le filtrage est traité dans le sous-chapitre suivant.

Conditionnement et filtrage (lignes 'I' et 'O')

INTRODUCTION

Le conditionnement autorise le déclenchement de l'extraction d'une Table (Entrée).

Le filtrage sélectionne des occurrences-résultats d'extraction (Sortie).

CONDITIONNEMENT : LIGNES DE TYPE 'I'

Le conditionnement est particulièrement utile lorsque l'extraction doit poursuivre son investigation à partir des données remplissant la ou les conditions. Cette investigation ne serait pas pertinente sur les occurrences exclues par le conditionnement.

Cette condition s'exprime sur une ligne de type 'I', en langage COBOL, sous la forme opérateur booléen + expression. Elle fait référence à une table hiérarchiquement supérieure déjà extraite.

EXEMPLE : On veut connaitre les occurrences utilisées par un Programme (extracteur "chevelu").

Pour l'entité Etat, on veut en extraire les occurrences si et seulement si l'appel de leur Structure de Données dans le -CD du Programme est de type 'I' ou 'J'.

```
NLG : L T REF. SELECTION
010 : 1 S PGM  P
020 : 2 S LSDO P PGM-CPGM CD
030 : 3 S LETA D LSDO-CSDO LR
040 :   I      LSDO-OSDOUT = 'I' OR 'J'
```

FILTRAGE : LIGNES DE TYPE 'O'

On peut être amené à vouloir filtrer un résultat d'extraction, c'est-à-dire à restreindre le nombre d'occurrences extraites.

Le filtrage s'exprime sur une ligne de type 'O' en langage PAF-SQL spécifique à la clause WHERE de l'ordre EXEC PAF DECLARE.

La syntaxe de ce langage est décrite dans ce manuel, chapitre "Fonctionnement et Utilisation dans les Programmes", sous-chapitre "Syntaxe du Langage PAF-SQL", Paragraphe "Déclaration d'un curseur".

Contrairement au conditionnement, le filtrage utilise comme critère une colonne de la Table courante.

NOTE : Au moins une ligne de filtrage est obligatoire après une ligne de sélection de type 'A'.

EXEMPLE : On veut obtenir la liste des Textes appelés dans un Rapport dont le code occurrence est RAP001. On souhaite également obtenir la liste des Rubriques chaînées à ces Textes.

```
NLG : L T REF. SELECTION
010 : 1 S TXT  T
020 : 2 S XRAP T TXT-CTXT XV
030 :   O      CRAP = 'RAP001'
040 : 3 S TXTD T XRAP-CTXT D
050 :   O      CRUB <> SPACE
```

Ventilation (ligne 'V')

Au cours des différentes boucles de traitement d'une extraction, certaines occurrences peuvent être extraites plusieurs fois. Par exemple, une Rubrique peut être utilisée plusieurs fois par un même Programme.

L'extraction de ces doubles peut être pertinente ou non. Dans le cas où l'on souhaite éliminer ces doubles, on devra mettre en oeuvre le principe de ventilation.

La ventilation associe à une occurrence un critère d'identification différent de celui défini par l'intermédiaire du cheminement (profondeur de l'extraction).

EXEMPLE : On cherche à connaitre les utilisations des Rubriques dans les Programmes. Chaque occurrence se verra identifiée par les critères suivants, selon la profondeur de l'extraction :

```
1  Programme       (PPPPPP) => PGM PPPPPP
2  Str. de Données (SD)     => PGM PPPPPP LSDO SD
3  Segment         (SDSS)   => PGM PPPPPP LSDO SD LSEG SDSS
4  Rubrique        (RRRRRR) => PGM PPPPPP LSDO SD LSEG SDSS
                               LRUB RRRRRR
```

Si on veut rattacher les Rubriques directement au Programme qui les utilise, on a besoin de définir un code curseur virtuel, ou curseur de ventilation (RUBP) dont le niveau d'imbrication est 2, puisqu'on passe ainsi du Programme à ses Rubriques :

```
=> PGM PPPPPP RUBP RRRRRR
```

VARIANTE : Si on souhaite rattacher les Rubriques à leur Structure de Données, le code curseur virtuel sera RUBD de niveau d'imbrication 3.

```
=> PGM PPPPPP LSDO SD RUBD RRRRRR
```

Ce curseur de ventilation est positionné à l'intérieur de la sélection (ligne de type 'S', 'I', ou 'O') de l'entité à ventiler, dans les quatre premiers caractères de la zone VEN.VIEW.

Ensuite, à la fin du PTex, on crée une ligne de type 'V' dont la colonne 'L' contient le niveau d'imbrication propre au curseur de ventilation. Ce curseur est renseigné dans la zone REF.

```
NLG : L T REF. SELECTION                    VEN.VIEW
010 : 1 S PGM  P
020 : 2 S LSDO P PGM-CPGM CD
030 : 3 S LSEG D LSDO-CSDO LS
040 :   I      (LSDO-OSDOUT NOT = 'I' AND 'J'
050 :   I       AND LSDO-OSDOOR = 'V' OR 'S')
060 : 4 S LRUB S LSEG-CSEG CE               RUBP
070 :   O      CRUB <> FILLER
900 : 2 V RUBP

VARIANTE :

NLG : L T REF. SELECTION                    VEN.VIEW
... : . . ..........................
030 : 3 S LSEG D LSDO-CSDO LS
... : . . ..........................
060 : 4 S LRUB S LSEG-CSEG CE               RUBD
070 : . . ..........................
900 : 3 V RUBD
```

Un curseur de ventilation peut (voire doit) être renseigné plusieurs fois si plusieurs branches du cheminement aboutissent à une même entité à partir d'une même entité de référence.

Par exemple, il existe plusieurs cheminements possibles pour obtenir les Rubriques utilisées par un Programme.

Pour reprendre l'exemple précédent, on utiliserait donc plusieurs fois le curseur RUBP, tous rattachés à une seule ligne 'V'.

REMARQUE : La zone SELECTION d'une ligne de type 'V' peut contenir un titre, un libellé, à titre documentaire.

Mise en forme (ligne 'P')

Les lignes de présentation (ou mise en forme) permettent de :

- Sélectionner des colonnes décrivant les occurrences résultats d'extraction,
- Spécifier une présentation particulière de ces sélections.

Les lignes 'P' peuvent être utilisées dans la description d'un Extracteur Utilisateur ou d'une Macro-Commande.

Les lignes 'P' ne sont pas obligatoires. Qu'elles soient saisies ou non, un PTEx crée toujours un fichier résultat 'brut' qui contient toutes les tables extraites au cours du cheminement (sauf celles déclarées non-éditables par l'utilisateur ; voir chapitre "PTEx : Ecrans et zones de définition et description", Rubrique "Options de Traitement", colonne 'O' "Options d'éditions".)

UTILISATIONS

Pour les Extracteurs Utilisateur, une ligne 'P' permet par exemple d'obtenir des listes personnalisées, de constituer automatiquement des commandes de procédures batch telles que GPRT ou EXTR (mais ceci est inutile pour UPDP).

Pour les Macro-Commandes, une ligne 'P' permet de spécifier une mise en forme qui sera reprise dans un rapport GDP+. Une mise en forme est appelée dans un Plan Type d'Edition par une ligne de type 'G'.

SYNTAXE DES LIGNES DE TYPE 'P' (Zone SELECTION)

La mise en forme s'exprime dans un langage positionnel, composé de la juxtaposition des paramètres suivants :

PPP,LLL,contenu

PPP : Positionnement :

Numérique, 3 caractères maximum, indique la position de début de transfert dans la zone réceptrice.

LLL : Longueur :

Numérique, 3 caractères maximum, indique la longueur de la zone à transférer dans la zone réceptrice.

contenu : Zone à transférer:

- Constante (bornée par la valeur présente dans la colonne 'V' de la zone OPDV, '}' par défaut),
- Colonne de la Table (code colonne) extraite par le curseur courant (généralement une colonne qui contient un code occurrence).
- Colonne d'une Table (curseur-colonne) extraite par un autre curseur.

  La possibilité d'appeler dans le "contenu" une colonne d'une autre Table permet de faire figurer dans une même mise en forme des colonnes issues de tables différentes.

'PPP,LLL,contenu' peut être répété autant de fois que nécessaire, sachant que la longueur expansée de la zone réceptrice ne doit pas dépasser la valeur spécifiée dans la zone TAILLE MAXIMALE dans la Définition du PTEx.

REMARQUE : Une même mise en forme peut être décrite sur plusieurs lignes 'P' consécutives si le curseur de présentation n'est saisi que sur la première de ces lignes. Voir le paragraphe "Positionnement des Curseurs de Présentation" ci-après pour plus de détails.

Pour une occurrence extraite, il est possible de définir plusieurs mises en forme et de sélectionner l'une ou plusieurs d'entre elles. Par exemple, une occurrence de type E correspond à la fois à une Rubrique et à une Propriété. L'utilisateur peut donc spécifier une certaine mise en forme pour une Rubrique et une autre pour une Propriété.

Il est possible de conditionner une ligne 'P' en la faisant suivre directement par une ligne 'I'.

```
Ex :  P  MEF1  1,8,'RUBRIQUE',10,6,CRUB
      I         RUB-TRUB = 'R'
      P  MEF2  1,9,'PROPRIETE',11,6,CRUB
      I         RUB-TRUB = 'I'
```

Si une mise en forme est associée à différents curseurs, le code curseur doit être remplacé par un astérisque '*'.

```
Ex :  P  MEF1  1,8,'RUBRIQUE',10,6,CRUB
      I        *-TRUB = 'R'
```

La virgule est la valeur par défaut pour délimiter les paramètres des lignes 'P'. Si l'on souhaite utiliser un autre caractère, il convient de le spécifier sous la colonne 'D' de la zone OPDV sur chaque ligne 'P' concernée.

SYNTAXE SPECIFIQUE DES LIGNES 'P' DES MACRO-COMMANDES

La zone SELECTION de la description d'une Occurrence d'Entité Utilisateur accepte un paramètre de présentation GDP '$VF', correspondant au caractère utilisé pour les séparateurs verticaux. La mise en forme est limitée à 132 caractères (pour éviter des troncatures lors de l'édition).

La totalité ou une partie des mises en forme d'un curseur peuvent être intégrées dans un rapport GDP+. Cela sera spécifié sur la description du Plan Type d'Edition.

>>>> Pour plus de détails, voir le manuel de Référence "Gestion de Documentation Personnalisée".

POSITIONNEMENT DES CURSEURS DE PRESENTATION

Un curseur de présentation est positionné à l'intérieur de la Table à mettre en forme, dans les quatre derniers caractères de la zone VEN.VIEW.

NOTE : S'il existe une ligne de ventilation (type 'V') pour un curseur à mettre en forme, il est recommandé de positionner le curseur directement sur cette ligne (voir la Variante 2 de l'exemple, ligne 900, ci-dessous).

Ensuite, la ligne 'P' est saisie, soit :

- Directement sous la sélection de la Table à mettre en forme, auquel cas l'indication du curseur de présentation dans la zone REF. est inutile, cf. Variante 2.
- A la fin du PTEx, cf. Variante 1.

  La zone REF. doit contenir le curseur de présentation précédemment renseigné, sauf dans le cas de la Variante 2.

On écrira autant de lignes 'P' qu'il y a de mises en forme nécessaires. Une mise en forme peut également être formulée sur plusieurs lignes 'P' (voir la REMARQUE ci-dessus).

A l'inverse, on peut associer une même mise en forme à un seul curseur.

```
NLG : L T REF. SELECTION                    VEN.VIEW
010 : 1 S PGM  P
020 : 2 S LSDO P PGM-CPGM CD
030 : 3 S LSEG D LSDO-CSDO LS
040 :   I      (LSDO-OSDOUT NOT = 'I' AND 'J'
050 :   I       AND LSDO-OSDOOR = 'V' OR 'S')
060 : 4 S LRUB S LSEG-CSEG CE
070 :   O      CRUB <> FILLER
080 :   P      2,7,}W1EX UE},9,6,CRUB
```

```
VARIANTE 1:
```

```
NLG : L T REF. SELECTION                    VEN.VIEW
... : . . .... ............................
060 : 4 S LRUB S LSEG-CSEG CE                   PRUB
070 :   O      CRUB <> FILLER
... : . . .... ............................
999 :   P PRUB 2,7,}W1EX UE},9,6,CRUB
```

```
VARIANTE 2:
```

```
NLG : L T REF. SELECTION                    VEN.VIEW
... : . . .... ............................
060 : 4 S LRUB S LSEG-CSEG CE               RUBP
... : . . .... ............................
900 : 2 V RUBP VENTILATION & SUPPORT M en F      PRUB
910 :   P      2,7,}W1EX UE},9,6,CRUB
```

On obtient ainsi des lignes de commande d'extraction de Rubriques, prêtes à être traitées par la procédure batch d'extraction EXTR.

Chapitre 10. Mise en oeuvre de PAF+

Entrées Utilisateur : PTEx Type E

Les entrées utilisateur sont les suivantes:

- Plusieurs lignes '*' - une obligatoire - identifiant l'utilisateur, la bibliothèque, et la session.

POS.	LON.	VALEUR	SIGNIFICATION
2	1	*	Type de ligne
3	8	uuuuuuuu	Code utilisateur
11	8	pppppppp	Mot de passe
19	3	bbb	Code bibliothèque
22	4	ssss	Numéro de session. Valeur par défaut: session courante
26	1		Etat de la session si historisée:
		' '	Session historisée
		T	Session historisée de test
27	1		Option de prise en compte du réseau de bibliothèques
		C	Valeur par défaut : Bibliothèque selectionnée et Bibliothèques supérieures. Voir le Guide de l'interface Utilisateur Mode caractères ou l'aide en ligne de Developer workbench pour les autres valeurs possibles

- Deux lignes, 'X' et 'Y', obligatoires. Elles sont utilisées, entre autres, pour préciser le domaine de l'extraction.

  Plusieurs lignes peuvent être utilisées pour paramétrer différentes exécutions d'un même Extracteur Utilisateur.

  – Une ligne 'X'

POS.	LON.	VALEUR	SIGNIFICATION
2	1	X	Type de ligne
3	4		Code curseur. Valeur par défaut : 1er curseur
7	32		Code occurrence. Valeur par défaut : toutes les occurrences. L'utilisation du caractère générique '*' est permise.

POS.	LON.	VALEUR	SIGNIFICATION
39	1	1	Impression d'un état debug. Valeur/défaut : pas d'état debug
40	6		Nombre d'enregistrements SYSPAF. Equivalent au paramètre SIZE / PAF. Valeur par défaut : 10

– Une ligne 'Y'

POS.	LON.	VALEUR	SIGNIFICATION
2	1	Y	Type de ligne
3	32		Borne de début
35	32		Borne de fin

Pour les informations techniques concernant cette exécution, voir le manuel "Les Procédures du Développeur", chapitre "Extraction personnalisée / Documentation automatisée", sous-chapitre "Extractions personnalisées - PAF+", section "XPAF - Validation d'un Plan Type d'Extraction".

Exemples PTEx et compte-rendu validation

Les pages qui suivent contiennent deux exemples de compte-rendu de validation de PTEx (Procédure XPAF).

Un compte-rendu XPAF est composé :

1. d'une page "COMMENTAIRE OU ERREUR",
2. de l'entrée utilisateur,
3. des lignes de Description du PTEx,
4. de la simulation de l'extraction.

Le premier exemple de PTEx explore les chaînages d'une Rubrique dans la Base VA Pac.

Le deuxième exemple exlore les chaînages propres à la Station de Travail autour de l'entité Dessin de Chaîne.

Exemple d'exécution d'un extracteur PAF+/CICS

```
000100 //*************************************************************
000120 //* VISUALAGE PACBASE
000130 //*
001100 //*               - JCL EXAMPLE -
001200 //*    USER EXTRACTOR SUBMISSION JCL
001300 //*    ADD A '*' LINE    (USER, PASSWORD, LIBRARY)
```

```
001400 //*    ****  ADD A X-TYPE LINE (REFER TO THE PAF USER REF.)
001500 //*               COL. 2    :   X
001600 //*               COL. 3    :   PAF CURSOR
001700 //*               COL. 7    :   UEO CODE
001800 //*               COL. 39   :   DEBUG TYPE (0 OU 1)
001900 //*               COL. 40   :   NUMBER OF PAF BUFFERS
002000 //*    ****  ADD A Y-TYPE LINE
002100 //*               COL. 3    :   START LIMIT
002200 //*               COL. 35   :   END LIMIT
002210 //*    CHANGE DCB OF PAC7SQ IN RELATION WITH THE USER ENTITY
002220 //*
002300 //*****************************************************************
002400 //PTEXJCL PROC BASE=$BASE,                CODE OF VAPAC DATABASE
002600 //      INDSV='$INDSV',                INDEX OF VSAM SYSTEM FILES
002700 //      INDSN='$INDSN',            INDEX OF NON-VSAM SYSTEM FILES
002800 //      INDUV='$INDUV',                 INDEX OF VSAM USER FILES
002900 //*:    VSAMCAT='$VCAT',                      USER VSAM CATALOG
003000 //*:    SYSTCAT='$SCAT',                    SYSTEM VSAM CATALOG
003100 //      STEPLIB=,                  USER LIBRARY OF LOAD-MODULES
003200 //      SORTLIB=,                                 SORT LIBRARY
003300 //      OUT=$OUT,                                 OUTPUT CLASS
003400 //      UWK=$UWK,                                    WORK UNIT
003500 //      SPAMB='(TRK,(1,1),RLSE)',             REQUESTS FILE SPACE
003600 //      SPASQ='(TRK,(5,1),RLSE)',               RESULT FILE SPACE
003700 //      SPAWK='(TRK,(5,1),RLSE)',                  WORK FILE SPACE
003800 //      CYL='(3,1)'                                 SORTWORK SPACE
005000 //*****************************************************************
005100 //INPUT  EXEC PGM=BVPTU001
005200 //**********************
005300 //STEPLIB  DD DSN=&STEPLIB,DISP=SHR
005302 //         DD DSN=$BCOB,DISP=SHR
005400 //CARTE    DD DDNAME=SYSIN
005500 //PAC7MB   DD DSN=&&XPAFMB,DISP=(,PASS),
005600 //          UNIT=&UWK,
005700 //          SPACE=&SPAMB,DCB=BLKSIZE=3440
005800 //MAXKEY  EXEC PGM=IDCAMS
005900 //**********************
006000 //*:STEPCAT DD DSN=&VSAMCAT,DISP=SHR
006100 //SYSPRINT DD SYSOUT=&OUT
006200 //SYSPAF   DD DSN=&&SYSPAF,DISP=(NEW,KEEP),
006300 //         SPACE=(CYL,(3,3)),
006400 //         LRECL=1031,RECORG=KS,KEYOFF=0,KEYLEN=12
006500 //MAXKEY   DD DSN=&INDSN..BVPSY(MAXKEY),DISP=SHR
006600 //SYSIN    DD DSN=&INDSN..BVPSY(REPRO999),DISP=SHR
006700 //*
006800 //-------  EXEC PGM=------
006900 //**********************
007000 //STEPLIB  DD DSN=&STEPLIB,DISP=SHR
007002 //         DD DSN=$BCOB,DISP=SHR
007100 //SORTLIB  DD DSN=&SORTLIB,DISP=SHR
007200 //*:STEPCAT  DD DSN=&VSAMCAT,DISP=SHR
007300 //PAC7AN   DD DSN=&INDUV..&BASE.AN,DISP=SHR
007400 //PAC7AR   DD DSN=&INDUV..&BASE.AR,DISP=SHR
007402 //PACGGN   DD DSN=&INDSV.BVPGN,DISP=SHRP=SHR
007404 //PACGGR   DD DSN=&INDSV.BVPGR,DISP=SHRP=SHR
```

```
007406 //PACGGU   DD DSN=&INDSV.BVPGU,DISP=SHRP=SHR
007500 //PAC7AE   DD DSN=&INDSV..BVPAE,DISP=SHR
007600 //SYSPAF   DD DSN=&&SYSPAF,DISP=(OLD,KEEP)
007700 //PAC7MB   DD DSN=&&XPAFMB,DISP=(OLD,DELETE)
007800 //PAC7SO   DD SPACE=&SPAWK,DCB=BLKSIZE=13080,
007900 //         UNIT=&UWK
008000 //PAC7SQ   DD DSN=&&PAC7SQ,DISP=(,PASS),
008100 //         UNIT=&UWK,
008200 //         DCB=(RECFM=FB,LRECL=80,BLKSIZE=6080),
008300 //         SPACE=&SPASQ
008400 //PAC7DB   DD SYSOUT=&OUT
008500 //SORTWK01 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
008600 //SORTWK02 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
008700 //SORTWK03 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
008800 //SYSOUT   DD SYSOUT=&OUT
008900 //SYSUDUMP DD SYSOUT=&OUT
009000 //SYSPRINT DD SYSOUT=&OUT
009100 //         PEND
010000 //PTEXJCL EXEC PTEXJCL
010200 //
```

Exemple d'exécution d'un extracteur PAF+/IMS

```
000100 //*************************************************************
000120 //* VISUALAGE PACBASE
000130 //*
001000 //* INSTALLATION  - JCL EXAMPLE -
001100 //*    USER EXTRACTOR SUBMISSION JCL
001200 //*    IN INPUT, ADD A '*' LINE    (USER, PASSWORD, LIBRARY)
001300 //*    ********  ADD A X-TYPE LINE (REFER TO THE PAF USER REF. )
001400 //*               COL. 2    :   X
001500 //*               COL. 3    :   PAF CURSOR
001600 //*               COL. 7    :   UEO CODE
001610 //*               COL. 39   :   DEBUG TYPE (0 OU 1)
001620 //*               COL. 40   :   NUMBER OF PAF BUFFERS
001650 //*    ********  ADD A Y-TYPE LINE
001700 //*               COL. 3    :   START LIMIT
001800 //*               COL. 35   :   END LIMIT
002100 //*    CHANGE DCB OF PAC7SQ IN RELATION WITH THE USER ENTITY
002200 //*************************************************************
002300 //PTEXJCL PROC BASE=$BASE,                CODE OF VAPAC DATABASE
002400 //      INDSV='$INDSV',               INDEX OF VSAM SYSTEM FILES
002500 //      INDSN='$INDSN',            INDEX OF NON-VSAM SYSTEM FILES
002600 //      INDUV='$INDUV',                  INDEX OF VSAM USER FILES
002700 //*:    VSAMCAT='$VCAT',                       USER VSAM CATALOG
002800 //*:    SYSTCAT='$SCAT',                     SYSTEM VSAM CATALOG
002900 //      STEPLIB=,                    USER LIBRARY OF LOAD-MODULES
003000 //      SORTLIB=,                                 SORT LIBRARY
003100 //      OUT=$OUT,                                 OUTPUT CLASS
003200 //      UWK=$UWK,                                    WORK UNIT
003300 //      SPAMB='(TRK,(1,1),RLSE)',             REQUESTS FILE SPACE
003400 //      SPASQ='(TRK,(5,1),RLSE)',               RESULT FILE SPACE
003500 //      SPAWK='(TRK,(5,1),RLSE)',                 WORK FILE SPACE
003600 //      CYL='(3,1)',                               SORTWORK SPACE
```

```
003700 //          PSBLIB='$PSBLIB',    IMS PSBLIB
003800 //          DBDLIB='$DBDLIB',    IMS DBDLIB
003900 //          RESLIB='$RESLIB',    IMS RESLIB
004000 //          PROCLIB='$PRCLIB',   IMS WORKLIB
004100 //         BUF=40,SPIE=0,TEST=0,EXCPVR=0,RST=0,PRLD=,SRCH=0,
004200 //         CKPTID=,MON=N,LOGA=0,FMTO=T,DBRC=$DBRC,IRLM=$IRLM
004300 //*************************************************************
004400 //INPUT  EXEC PGM=BVPTU001
004500 //**********************
004600 //STEPLIB  DD DSN=&STEPLIB,DISP=SHR
004700 //         DD DSN=$BCOB,DISP=SHR
004800 //CARTE    DD DDNAME=SYSIN
004900 //PAC7MB   DD DSN=&&XPAFMB,DISP=(,PASS),
005000 //          UNIT=&UWK,
005100 //          SPACE=&SPAMB,DCB=BLKSIZE=3440
005200 //*
005300 //______  EXEC PGM=DFSRRC00,REGION=$REGSIZ,
005400 //        PARM=(DLI,______,______,&BUF,
005500 //        &SPIE&TEST&EXCPVR&RST,&PRLD,
005600 //        &SRCH,&CKPTID,&MON,&LOGA,&FMTO,,,&DBRC,&IRLM)
005700 //STEPLIB  DD DSN=&RESLIB,DISP=SHR
005800 //         DD DSN=&STEPLIB,DISP=SHR
005900 //         DD DSN=$BCOB,DISP=SHR
006000 //DFSRESLB DD DSN=&RESLIB,DISP=SHR
006100 //IMS      DD DSN=&PSBLIB,DISP=SHR
006200 //         DD DSN=&DBDLIB,DISP=SHR
006300 //*:STEPCAT DD DSN=&SYSTCAT,DISP=SHR
006400 //*:       DD DSN=&VSAMCAT,DISP=SHR
006500 //SYSOUT   DD SYSOUT=&OUT
006600 //SYSOUX   DD SYSOUT=&OUT
006700 //DDSNAP   DD SYSOUT=&OUT
006800 //PROCLIB  DD DSN=&PROCLIB,DISP=SHR
006900 //IEFRDER  DD DUMMY,
007000 //            DCB=(RECFM=VB,BLKSIZE=1920,LRECL=1916,BUFNO=2)
007100 //SYSUDUMP DD SYSOUT=&OUT,DCB=(RECFM=FBA,LRECL=121,
007200 //            BLKSIZE=605),SPACE=(605,(500,500),RLSE,,ROUND)
007300 //IMSUDUMP DD SYSOUT=&OUT,DCB=(RECFM=FBA,LRECL=121,
007400 //            BLKSIZE=605),SPACE=(605,(500,500),RLSE,,ROUND)
007500 //IMSMON   DD DUMMY
007600 //DFSVSAMP DD DSN=&INDSN..BVPSY(DFSVSAM8),DISP=SHR
007700 //SORTLIB  DD DSN=&SORTLIB,DISP=SHR
007800 //SORTWK01 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
007900 //SORTWK02 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
008000 //SORTWK03 DD UNIT=&UWK,SPACE=(CYL,&CYL,,CONTIG)
008100 //BVP7AE   DD DSN=&INDSV..BVPAE,DISP=SHR
008200 //B7AN$BASE   DD DSN=&INDUV..&BASE.AN,DISP=SHR
008300 //B7AR$BASE   DD DSN=&INDUV..&BASE.AR,DISP=SHR
008310 //BVP7GN   DD DSN=&INDSV..BVPGN,DISP=SHR
008320 //BVP7GR   DD DSN=&INDSV..BVPGR,DISP=SHR
008330 //BVP7GU   DD DSN=&INDSV..BVPGU,DISP=SHR
008400 //B7PA$BASE   DD DSN=&INDUV..&BASE.PA,DISP=SHR
008500 //B7P1$BASE   DD DSN=&INDUV..&BASE.P1,DISP=SHR
008600 //PAC7MB   DD DSN=&&PAC7MB,DISP=(OLD,DELETE)
008700 //PAC7SO   DD UNIT=&UWK,SPACE=&SPAWK,DCB=BLKSIZE=27904
008800 //PAC7SQ   DD DSN=&&PAC7SQ,DISP=(,PASS),UNIT=&UWK,
```

```
008900 //            DCB=(RECFM=FB,LRECL=80,BLKSIZE=6080),
009000 //            SPACE=(&SPASQ,RLSE)
009100 //PAC7DB   DD SYSOUT=&OUT
009200 //*
```

Exemple d'exécution d'un extracteur PAF+/WNT

```
000010 REM * -------------------------------------------------------
000020 REM *      VISUALAGE PACBASE
000030 REM *
000040 REM * -------------------------------------------------------
000050 REM *         - EXAMPLE OF PAF+ PROCEDURE EXECUTION -
000060 REM *
000070 REM * -------------------------------------------------------
000080 REM *  INPUT FILE DESCRIPTION :
000090 REM *  ADD A '*' LINE    (USER, PASSWORD, LIBRARY)
000100 REM *  ****  ADD A X-TYPE LINE (REFER TO THE PAF USER REF.)
000110 REM *             COL. 2    :   X
000120 REM *             COL. 3    :   PAF CURSOR
000130 REM *             COL. 7    :   UEO CODE
000140 REM *             COL. 39   :   DEBUG TYPE (0 OU 1)
000150 REM *             COL. 40   :   NUMBER OF PAF BUFFERS
000160 REM *  ****  ADD A Y-TYPE LINE
000170 REM *             COL. 3    :   START LIMIT
000180 REM *             COL. 35   :   END LIMIT
000190 REM *  CHANGE DCB OF PAC7SQ IN RELATION WITH THE USER ENTITY
000200 REM *
000210 REM *
000800
000900 Set WshShell = Wscript.CreateObject("Wscript.Shell")
001000 Set Args = Wscript.Arguments
001100 Set WshEnv = WshShell.Environment("PROCESS")
001200 Set WshVolEnv = WshShell.Environment("VOLATILE")
001300
001400 Dim FSO
001500 Set  FSO = CreateObject("Scripting.FileSystemObject")
001600 Dim ObjProcess
001700 Set  ObjProcess = CreateObject("BvpWsh.Process")
001800
001900 Rep_BVP="HKLM\SOFTWARE\IBM\BVP VisualAge Pacbase\Server\BVP"
002000
002100 Rep_Proc = WshShell.RegRead(Rep_BVP & "_SYS\PROC\")
002200 Rep_NLS = WshShell.RegRead (Rep_BVP & "_SYS\NLS\")
002300 Rep_SKEL = WshShell.RegRead (Rep_BVP & "_SYS\SKEL\")
002400 Rep_PGM = WshShell.RegRead (Rep_BVP & "_SYS\PGM\")
002500
002600   WshEnv("PATH") = Rep_PGM & ";" & WshEnv("PATH")
002700
002800 ' Set COBOL environments :
002900 ' ------------------------
003000 Var = "HKLM\SOFTWARE\Micro Focus\NetExpress\"
003100
003200 On Error Resume Next
003300 NetExpress_Version = WshShell.RegRead (Var & _
```

```
003400                        "DefaultVersion")
003500 If Err.Number = 0 Then
003600    NetExpress = 1
003700 Else
003800    NetExpress = 0
003900 End if
004000
004100 On Error Resume Next
004200 NetExpress_Version = WshShell.RegRead (Var & "Version")
004300 If Err.Number = 0 Then
004400    ApplicationServer = 1
004500 Else
004600    ApplicationServer = 0
004700 End if
004800
004900
005000 If NetExpress = 1 or ApplicationServer = 1 Then
005100    Var = Var & NetExpress_Version & "\COBOL\"
005200    COBOL_Version = WshShell.RegRead (Var & "Version")
005300    Var = Var & COBOL_Version & "\Environment\"
005400    COBPATH =  WshShell.RegRead (Var & "PATH")
005500
005600    WshEnv("COBDIR") = WshShell.RegRead (Var & "COBDIR")
005700    WshEnv("COBHNF") = WshShell.RegRead (Var & "COBHNF")
005800    WshEnv("PATH") = COBPATH & ";" & WshEnv("PATH")
005900 Else
006000  wscript.echo "NetExpress or Application Server
006100                              not implemented. "
006200 End if
006300 '============================================================
006400
006500 ' EXECUTION of PAFPGP
006600 '-----------------------------------
006700
006800 ' 1-Delete PAF File
006900 If  FSO.FileExists(Rep_TMP & "\WPAF.tmp") Then
007000 Set FileD = FSO.GetFile(Rep_TMP & "\WPAF.tmp")
007100 FileD.Delete
007200 End If
007300
007400 ' 2-Assign database files ...
007500 WshEnv("PAC7AE") = Rep_SKEL & "\AE"
007600 WshEnv("PAC7AR") = Rep_BASE & "\AR"
007700 WshEnv("PAC7AN") = Rep_BASE & "\AN"
007800 WshEnv("PACGGN") = Rep_ABASE & "\AN"
007810 WshEnv("PACGGR") = Rep_ABASE & "\AR"
007820 WshEnv("PACGGU") = Rep_ABASE & "\GU"
007840
007900 ' 2bis- ... and  PAF File
007910 WshEnv("SYSPAF") = Rep_TMP & "\WPAF.tmp"
007920
008000 ' 2ter- Input file for PAF
008010 WshEnv("PAC7MB") = ...
008100
008200 ' 3-Assign of program  files
```

```
008300 ' ......................
008400
008500 ' 4- Execute ...
008600 Return = WshShell.Run("BVPAFPGP.exe ", 1, TRUE)
008700
008800 ' 5- If Error
008900 if err.number <> 0 and Return = 0 then
009000  wscript.echo "Syntax or RunTime Error"
009100 End if
009200
009300 if  Return <> 0 then
009400  wscript.echo "PAFPGP execution error"
009500 End if
009600
009700
009800 Wscript.Quit (Return)
009900
```

Exemple d'exécution d'un extracteur PAF+/UNIX

```
000100 #!/bin/sh
000200 #
000300 # EXEMPLE OF AN EXECUTION PROCEDURE USING A PAF+ PROGRAM
000400 # ------------------------------------------------------
000500 # (execution of PGPAFP program)
000600
000700 # Input file description :
000800 # - one '*' LINE    (USER, PASSWORD, LIBRARY)
000900 # - ONE X-TYPE LINE (REFER TO THE PAF USER REF. )
001000 #   COL. 2    :   X
001100 #   COL. 3    :   PAF CURSOR
001200 #   COL. 7    :   UEO CODE
001300 #   COL. 39   :   DEBUG TYPE (0 OU 1)
001400 #   COL. 40   :   NUMBER OF PAF BUFFERS
001500 # - one Y-TYPE LINE
001600 #   COL. 3    :   START LIMIT
001700 #   COL. 35   :   END LIMIT
001750
001800 # Cobol compiler environment variable :
001900 COBDIR=/usr/local/cobol
002000 export COBDIR
002100
002200 # PATH environment variable :
002300 PATH=.:$COBDIR/bin:$PATH
002400 export PATH
002500
002600 # LIBPATH environment variable :
002700 # (for AIX version)
002800 LIBPATH=/usr/lib:$COBDIR/lib
002900 export LIBPATH
003000
003100 # LD_LIBRARY_PATH environment variable :
003200 # (for HP-UX, OSF1, SunOS or Linux version)
003300 LD_LIBRARY_PATH=/usr/lib:$COBDIR/lib
```

```
003400 export LD_LIBRARY_PATH
003500
003600 # VisualAge Pacbase directory environment variable :
003700 PACDIR="/lv00/ll350/paclanx"
003800 export PACDIR
003900
004000 # COBPATH environment variable :
004100 # (current directory + VisualAge Pacbase programs directory)
004200 COBPATH=.:$PACDIR/system/gnt
004300 export COBPATH
004400
004500 # VisualAge Pacbase database name (mandatory) :
004600 BASE=BVAP
004700
004800 # VisualAge Pacbase environment variable (mandatory) :
004900 . $PACDIR/config/$BASE/PAC7AE.ini
005000 . $PACDIR/config/$BASE/PAC7AN.ini
005100 . $PACDIR/config/$BASE/PAC7AR.ini
005200 . $PACDIR/config/$BASE/PACGGN.ini
005300 . $PACDIR/config/$BASE/PACGGR.ini
005400 . $PACDIR/config/$BASE/PACGGU.ini
005500 SYSPAF=./wpaf
005600 export SYSPAF
005700 PAC7SO=./pafp.so
005800 export PAC7SO
005900 PAC7SQ=./pafp.sq
006000 export PAC7SQ
006100 PAC7DB=./pafp.db
006200 export PAC7DB
006300
006400 # Transaction input file assignment (mandatory) :
006500 PAC7MB=./MBPAFP
006600
006700 # User application environment variable :
006800 FILE1=./file1
006900 export FILE1
007000
007100 # Start execution :
007200 cobrun PGPAFP
007300
007400 # Deletion of the temporary files :
007500 if [ -r "$SYSPAF" ]
007600 then
007700    rm $SYSPAF*
007800 fi
```

Référence : DDPAF000351F - 7509

Imprimé en France