Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER EN SCIENCES INFORMATIQUES

Conception de la base de données d'un système d'information d'un organisme bancaire

Collin, Benoit

Award date:
1981

Awarding institution:
Universite de Namur

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 02. May. 2026

FACULTES
UNIVERSITAIRES
N.D. DE LA PAIX

NAMUR

INSTITUT D'INFORMATIQUE



RUE GRANDGAGNAGE, 21, B - 5000 NAMUR (BELGIUM)

FACULTES
UNIVERSITAIRES
N.-D. DE LA PAIX
NAMUR

Bibliothèque
FMB16
1981/3

FM B16/1981/3

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX (NAMUR)

INSTITUT D'INFORMATIQUE

CONCEPTION DE LA BASE DE DONNEES D'UN SYSTEME D'INFORMATION D'UN ORGANISME BANCAIRE.

Mémoire présenté par

Benoît Collin

en vue de l'obtention
du titre de
Licencié et Maître en Informatique.

Année académique 1980-1981

LBS 3220810
200950

A ma femme, pour sa patience et sa collaboration.

REMERCIEMENTS

Je tiens , ici, à remercier Monsieur Jean-Luc Hainaut, promoteur de ce mémoire, pour sa disponiblité et ses conseils qui ont permis la réalisation de celui-ci.

Je remercie également Messieurs Hachez et Hans ainsi que Madame Deleeuw de la Banque Bruxelles-Lambert sans qui ce mémoire n'aurait pas vu le jour.

J'exprime , enfin, ma reconnaissance à Jean-Paul Adams et à Patrick Lambion pour leur collaboration lors de l'édition de ce mémoire.

TABLE DES MATIERES

TABLE DES MATIERES

- Introduction 2

- Première partie : Les outils et la démarche 4

  A Introduction 5
  B Le Modèle d'analyse conceptuelle 7
  C Le Modèle d'accès et son expression binaire 13
  D Transposition du modèle d'analyse conceptuelle en un modèle d'accès 18
  E Langage de description des applications 21

- Deuxième partie : Analyse conceptuelle du système 26

  A Introduction 27
  B Description sommaire du système CORRESPONDANTS 27
  C Schéma conceptuel du système 29
  D Description sémantique des entités 30
  E Expression sémantique et contraintes de cardinalité et d'existence des relations du schéma 34
  F Classification des propriétés 36
  G Enoncé des contraintes 38
  H Description des applications 40
  I Quantifications 46

- Troisième partie : Etude des accès 49

  A Introduction 50
  B Transformation du schéma conceptuel en un schéma des accès logiques 50
  C Analyse des applications 52

- Quatrième partie : Expression des structures de données et des primitives d'accès 74

  A Les Structures de données 75
  B Les primitives 87

- Cinquième partie : Propositions d'implémentation 100

  A Introduction 101
  B Enoncé des critères 102
  C Proposition d'implémentation pour les types d'article 102
  D Proposition d'implémentation pour les types de chemin 112

  - Conclusion 134

- Bibliographie

- Annexes
  - Annexe 1 : Précisions sur l'organisation physique des data sets, sets, et subsets
  - Annexe 2 : Schéma DDL de la base "correspondants"
  - Annexe 3 : Image des items par types d'article
  - Annexe 4 : Transformations de structures
  - Annexe 5 : Estimation du volume de la base

INTRODUCTION

INTRODUCTION

Ce mémoire consiste en l'étude de la base de données d'un système d'information développé à la banque BRUXELLES-LAMBERT : le système "correspondants". Les responsables du service "base de données" de la banque étaient en effet, intéressés par cette étude et ce, afin de pouvoir comparer la base qu'ils avaient obtenue avec celle obtenue par une démarche différente de la leur.

Définissons dans un premier temps le concept principal du système et la fonction essentielle de celui-ci. Un correspondant est le siège d'une banque situé dans une ville et qui entretient des relations avec la banque Bruxelles-Lambert soit directement, soit par l'intermédiaire d'autres organismes bancaires. La base du système se veut être le signalétique de ces correspondants, qui contiendra des informations sur leur raison sociale, leurs adresses (SWIFT, TELEX,...)et les comptes qu'ils possèdent ou peuvent utiliser.

Pour réaliser l'étude de ce système, il fallait nous définir une méthode. Sur le thème de l'analyse et de la conception de système d'information, la littérature était abondante. Nous n'aborderons cependant pas ici, le problème du choix de la méthode car, à nos yeux, une de celles-ci s'imposait : celle proposée à l'institut d'informatique où nous suivions les cours depuis deux ans. Nous comparerons cette méthode avec celles définies par H.TARDIEU et G.BENCI dans notre conclusion. Nous essayerons également de discerner certaines limites que nous avons rencontrées.

Dans la première partie de cet ouvrage, nous présenterons donc la démarche qui nous a guidé ainsi qu'un certain nombre d'outils méthodologiques qui la supportent. Ceux-ci sont des outils développés tels quels à l'institut ou dont le champ d'application et la généralité ont été restreints au cadre du problème qui nous occupait. Le lecteur ne doit cependant , en aucun cas, prendre cet ouvrage comme une référence exacte aux travaux effectués à l'institut. Pour ce faire, on se rapportera à la bibliographie de cette étude.

Les deuxième, troisième et cinquième parties constituent l'application de la démarche proposée au cas du système "correspondants". Chaque partie constitue un tout indépendant des étapes suivantes. La fonction de chaque partie étant de décrire le système en fonction de critères inconnus à l'étape antérieure et de façon cohérente avec celle-ci.

La quatrième partie nous servira à nous dégager des particularités du système de gestion de base de données retenu en caractérisant les structures de données et d'accès de celui-ci en fonction des structures du modèle dans lequel le système aura été décrit lors de la troisième partie.

Nous n'avons pu être toujours aussi rigoureux que ne le prescrit la méthode; nous aborderons ce problème dans notre conclusion. C'est pourquoi, au terme méthode, nous avons préféré celui de démarche qui propose un fil conducteur, un certain nombre d'étapes à réaliser dans un cadre précis.

PREMIERE PARTIE

Première partie : Les outils et la démarche

A. Introduction

Des différents travaux qui ont été entrepris depuis le début des années 70 , il ressort que l'un des acquis fondamentaux est à l'heure actuelle, la reconnaissance dans le processus d'analyse d'une base de données ou d'un système d'information , d'une hiérarchie de niveaux de description et par la même de conception.

La fonction d'un niveau de description comme étape d'une analyse hiérarchisée est de décrire un système en fonction de certains critères inconnus aux niveaux supérieurs mais de façon cohérente avec celui-ci. Cette notion de critères inconnus fait référence à l'un des objectifs de l'analyse en niveaux : un niveau est indépendant des critères pris en charge par les niveaux inférieurs.

La plupart des méthodes proposées actuellement retiennent ce principe. Elles diffèrent essentiellement sur la définition de ces niveaux et sur la manière de prendre en charge chacun de ceux-ci. La démarche que nous allons proposer ici, s'inspire largement des travaux effectués à l'institut d'informatique par Messieurs Bodart, Hainaut et Van Lamsweerde. On trouvera un exposé complet de cette méthode dans [1].

La première étape de notre démarche dénommée "analyse conceptuelle du système" permettra de formaliser le réel perçu dans un modèle que nous présenterons ci-dessous. Ce réel perçu contient toutes les informations que l'on veut mémoriser dans la base et les traitements que l'on effectuera sur celles-ci.

La deuxième étape dénommée "Phase d'analyse des accès logiques" permettra de définir le "cahier des charges" nécessaire à la mise en oeuvre sur machine réelle.

Dans un premier temps, nous représenterons le résultat de l'analyse conceptuelle : le schéma ,par un schéma d'accès aux données en utilisant un certain nombre de règles qui font partie des outils disponibles dans le cadre de la démarche. Ce schéma sera , lui, formalisé dans un autre modèle : le modèle d'accès que nous présenterons également. Il fournira tous les accès à toutes les données et contiendra sous une forme adéquate les contraintes d'intégrité du schéma conceptuel.

Dans le deuxième temps de cette étape, on établira la description statique des données décrites par le schéma des accès.

Pour chacun des traitements (applications) identifiés lors de l'analyse conceptuelle, on décrira un algorithme en termes d'accès à la base. Ces algorithmes nous permettront de mettre en évidence les composants du schéma des accès réellement utilisés. Ils permettront également de mettre en évidence les taux d'utilisation de chacun de ces composants. Les résultats quantitatifs obtenus dépendent de l'analyse conceptuelle et des algorithmes décrits. Ils ne sont donc pas liés à un machine réelle. Ils déterminent un trafic logique relatif à la base mais ne permettent pas d'en préciser le trafic physique.

La troisième étape nous permettra de nous dégager de la machine réelle (SGBD) pour ne retenir que les structures de données et les primitives d'accès qu'offre celui-ci. Nous emploierons de nouveau le modèle d'accès pour formaliser le ou les SGBD retenus.

Dans la dernière étape, enfin, nous proposerons diverses implémentations pour chacun des composants du schéma des accès identifiés lors de la deuxième étape. Nous analyserons ces composantes en fonction de certains critères que nous énoncerons à ce moment. Nous choisirons enfin, parmi les propositions, celle qui répondra le plus avantageusement aux critères proposés.

B. Le modèle d'analyse conceptuelle

Pour exprimer l'analyse conceptuelle qui est, nous le rappelons, une formalisation de réel perçu d'une organisation, d'un système d'information , nous avons retenu un modèle entité/relation. Nous allons d'abord en décrire les concepts, ensuite nous préciserons les différentes étapes de l'analyse conceptuelle et les différents formalismes employés pour exprimer celles-ci. Ce modèle est tiré de [16], [11].

1. Les concepts

Nous représenterons la structure d'une information par un modèle élaboré à partir des concepts.

- entité
- association
- propriété

a) une entité

C'est ce qu'un individu perçoit comme un tout ayant une existence propre. Une entité est caractérisée par un ensemble de propriétés quantitatives ou qualitatives

Exemple : employé

b) une relation

C'est un ensemble de deux entités ou chacune assume un rôle donné. Une relation peut elle-même posséder des propriétés. L'existence d'une relation est contingente à l'existence des entités qu'elle relate.

Exemple : -travaille- est une relation entre les entités employé et usine.

c) une propriété

Une propriété appartenant à une entité ou à une relation est une qualité qu'un individu attribue à cette entité ou à cette relation. L'existence d'une propriété est contingente à l'existence de l'entité ou de la relation concernée.

Exemple : numéro d'employé.

d)TYPE - OCCURRENCE

Nous rapellerons ici la distinction entre type et occurrence.

- Un type d'entité est l'ensemble des entités caractérisées par les mêmes propriétés.

- Un type de relation est l'ensemble des relations caractérisées par les mêmes propriétés.

- Un type de valeur est un ensemble de valeurs ou un produit d'ensembles de valeurs de même nature.

Une occurrence est un élément d'un ensemble.

e) Typologie des propriétés

On peut classer les propriétés associées aux entités ou aux relations :

- celles qui identifient l'entité , la relation ou non

- celles qui sont obligatoires (leur existence conditionne celle de l'entité ou de la relation)

- celles qui sont facultatives

- celles qui sont répétitives (plusieurs occurrences par entité, relation)

f) Contraintes d'intégrité

Les contraintes d'intégrité sont des prédicats relatifs aux données de la base et définis sur les éléments du schéma conceptuel, qui doivent être toujours vrais. Un certain nombre de ces contraintes peuvent être exprimées dans notre modèle; nous allons les décrire. Les contraintes restantes seront énoncées en français.

1) Contraintes de cardinalité

Nous pouvons pour exprimer celles-ci reprendre le formalisme des modèles binaires étant donné que l'on s'est limité aux relations binaires

relation a) (1-1)
b) (1-N)
c) (N-1)
d) (M-N)

Une relation (1-1) entre deux types d'entités A et B signifie qu'une occurrence du type A ne pourra être en relation qu'avec une seule entité du type B et réciproquement.

Une relation (1-N) entre deux types d'entités A et B signifie qu'une occurrence du type A pourra être en relation avec plusieurs entités du type B mais qu'une occurrence du type B ne sera jamais en relation qu'avec une seule du type A.

Une relation (N-1) entre deux types d'entités A et B signifie qu'une occurrence du type A pourra être en relation avec une seule occurrence du type B et que une occurrence du type B pourra être en relation avec plusieurs occurrences du type A.

Une relation (M-N) entre deux types d'entités A et B signifie qu'une occurrence de A pourra être en relation avec plusieurs occurrences du type B et qu'une occurrence de B pourra être en relation avec plusieurs occurrences du type A.

2) Contraintes d'existence

- Un type de relation est fort pour un type d'entité si une entité de celui-ci ne peut exister que si elle appartient à au moins une relation de ce type.

- Un type de relation est faible pour un type d'entité si l'existence d'une entité de ce type est indépendante de la relation.

2. Représentation graphique

Nous représenterons le schéma conceptuel en utilisant les conventions suivantes :

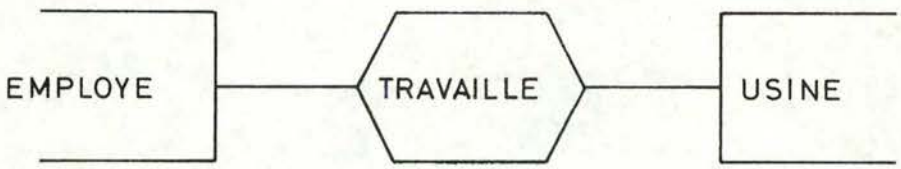
a) Un type d'entité sera représenté par son nom en caractères majuscules dans un rectangle. Dans celui-ci figurera également le nom des propriétés de ce type en caractères minuscules.

Exemple :



b) Un type de relation sera représenté par un hexagone non régulier dans lequel figurera son nom en caractères majuscules et celui de ses propriétés en caractères minuscules. Nous relierons le type de relation aux types d'entités qui la composent par des traits continus.

Exemple :



3. Expression sémantique des entités et des relations et expression des contraintes d'existence et de cardinalité.

Après avoir établi le schéma conceptuel , nous devons décrire la sémantique des entités et des relations

a) Expression sémantique des entités

Nous décrirons les types d'entité du schéma en prenant une occurrence de chaque type et en donnant sa signification ainsi que celle de ses propriétés Nous exprimerons l'appartenance d'une

propriété à une entité en utilisant la notation pointée.

Exemple :

Nous décrirons le type d'entité EMPLOYE et ses deux propriétés , nom-employé et numéro-employé par :

EMPLOYE {si e appartient à EMPLOYE ,e est un employé
dont le numéro identifiant au sein de l'usine
est représenté par (e.numéro-employé)
dont le nom est représenté par (e.nom-employé)}

(e.numéro-employé) représente le numéro de l'employé (e).

b) Expression sémantique et contraintes d'existence et de cardinalité des relations
---

Nous représenterons les types de relations par le nom entre parenthèses des types d'entité les composant . Nous les décrirons en prenant une occurrence de chaque type d'entité qui les compose et en donnant la signification de la relation qui les unit.

Nous exprimerons les contraintes de cardinalité sous la forme décrite plus haut (1-1, 1-N, N-1, M-N). Les contraintes d'existence seront représentées dans l'énoncé de la relation en soulignant le ou les types d'entité pour lesquelles la relation est forte.

Exemple :

TRAVAILLE (EMPLOYE,USINE) cardinalité N-1.
{(e,u) l'employé (e) travaille dans l'usine (u)}

4. Classification des propriétés

Ici, nous préciserons le type des propriétés en utilisant la typologie présentée ci-dessus. Nous réaliserons cette analyse par type d'entité et de relation

Exemple :

ENTITE EMPLOYE

-numéro employé identifiant, obligatoire
-nom employé obligatoire

5. Description des applications

Nous décrirons les applications du système en termes de données d'entrée, de données de sortie et de fonction. Nous donnerons également le nombre d'activations de ces fonctions par jour.

6. Quantifications

Dans un dernier temps , nous établirons la quantification des données : le nombre d'entités de chaque type, le nombre moyen d'entités cibles de chaque relation, les différentes probabilités. Nous ne regrouperons pas toutes ces quantifications dans l'analyse conceptuelle bien qu'elles en fassent partie, mais nous les présenterons chaque fois que nous en aurons besoin.

C. Le modèle d'accès et son expression binaire

Le modèle d'accès pourrait être perçu comme une généralisation et une simplification des propositions CODASYL. Il a été développé à l'institut d'informatique par J-L Hainaut . Il veut offrir à l'utilisateur une description de données qui soit simple, indépendante des langages de description de données et suffisamment précise pour permettre une exploitation efficace de ces bases de données. Les structures de données qu'il décrit et que nous allons présenter sont familières aux programmeurs d'application et aux analystes organiques. A ces structures sont associées les primitives qui permettent de manipuler les données ainsi décrites. Ce modèle va nous permettre d'exprimer le schéma conceptuel moyennant quelques transformations présentées ci-après en un schéma des accès. Il nous servira également à caractériser les structures de données admises par DMS II. On se réfèrera à [12] pour trouver une description complète de ce modèle.

Les objets

-L'élément de base est l'article, unité d'information enregistrée, qui peut faire l'objet d'accès, de création et de suppression. Tous les articles sont distincts et chacun appartient à un type d'article qui en définit les propriétés générales.

-Une valeur d'item est une donnée manipulable dans un programme et qui appartient à un type appelé item. A un article peut être associé des valeurs d'item. Pour un type d'article, un item peut être simple ou répétitif, obligatoire ou facultatif. Un item peut encore être décomposable en item élémentaire.

-Un fichier est une collection d'articles de types éventuellement différents, un article appartenant toujours à un fichier.

-Une base de données est la collection des articles de différents fichiers. Elle contient un article d'un type particulier, appelé système, qui constitue un point d'entrée privilégié.

-Un identifiant est un item (ou une liste d'items) d'un type d'article tel qu'il n'existe pas dans le référentiel précisé (ex : Type d'article), plus d'un article qui soit associé à une même valeur de cet item (ou des items de cette liste).

Les mécanismes d'accès

L'accès est pour un programme la mise à disposition d'un objet de la base de données.

Les différents types d'accès sont :

- l'accès à une base de données, représenté par l'accès à son article SYSTEME

- l'accès à un fichier

- l'accès à un article d'une base de données, d'un fichier

-l'accès aux valeurs d'item d'un article

- l'accès aux articles auxquels est associé une valeur déterminée d'un item (clé d'accès)

- l'accès à des articles à partir d'un article qui est appelé chemin d'accès.

Ce chemin d'accès est un mécanisme qui associe à un article dit origine 0,1 ou plusieurs articles dits cibles, d'une manière telle qu'il soit possible d'accéder à partir de l'article origine, successivement aux articles cibles. Tout chemin appartient à un et un seul type qui en définit les propriétés générales.

Un type de chemin est caractérisé par les types d'articles auxquels doivent appartenir d'une part, les articles origines et d'autre part, les articles cibles de ses chemins, par sa connectivité (M-N, 1-N, N-1, 1-1) de l'origine vers la cible ,par l'existence d'un chemin inverse.

Les primitives

Elles correspondent aux opérations élémentaires qu'il est possible d'effectuer sur les objets d'une base de données. Elles peuvent être rangées en trois classes : les primitives d'accès, les primitives de modification qui font évoluer l'état de la base de données et les primitives de contrôle d'environnement qui fonctionnellement permettent la création de super-primitives, l'établissement de points de reprise et la maîtrise de la concurrence.

Représentation graphique des structures du modèle d'accès

a) Type d'article.

Un type d'article est représenté par son nom entouré d'un rectangle



b) Item

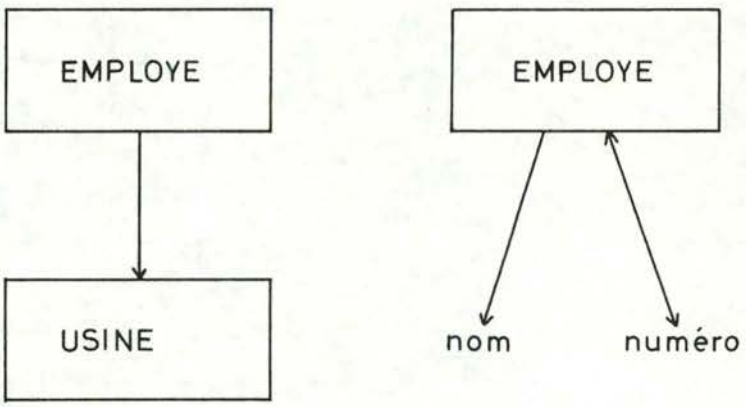
Un item est représenté par son nom.

NO-EMPL ou no-empl

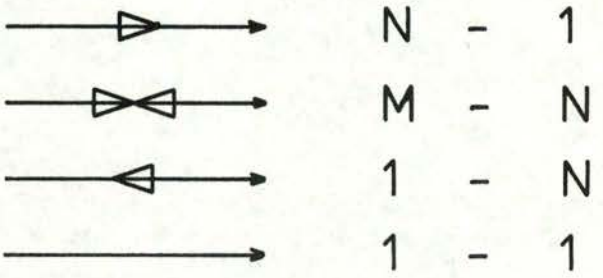
c) Type de chemins entre types d'articles ou entre type d'articles et item.

Un tel type de chemin est représenté par un arc orienté reliant les deux représentations :

a) deux types d'articles
b) un type d'article et un item

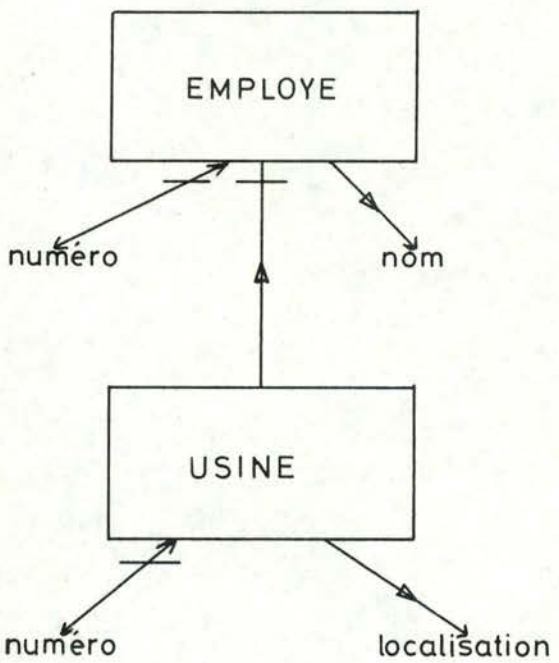


On précisera en outre le nom d'un type de chemin sur l'arc le représentant. Nous préciserons la cardinalité d'un type de chemin par un symbole placé sur l'arc le symbolisant avec les conventions suivantes :



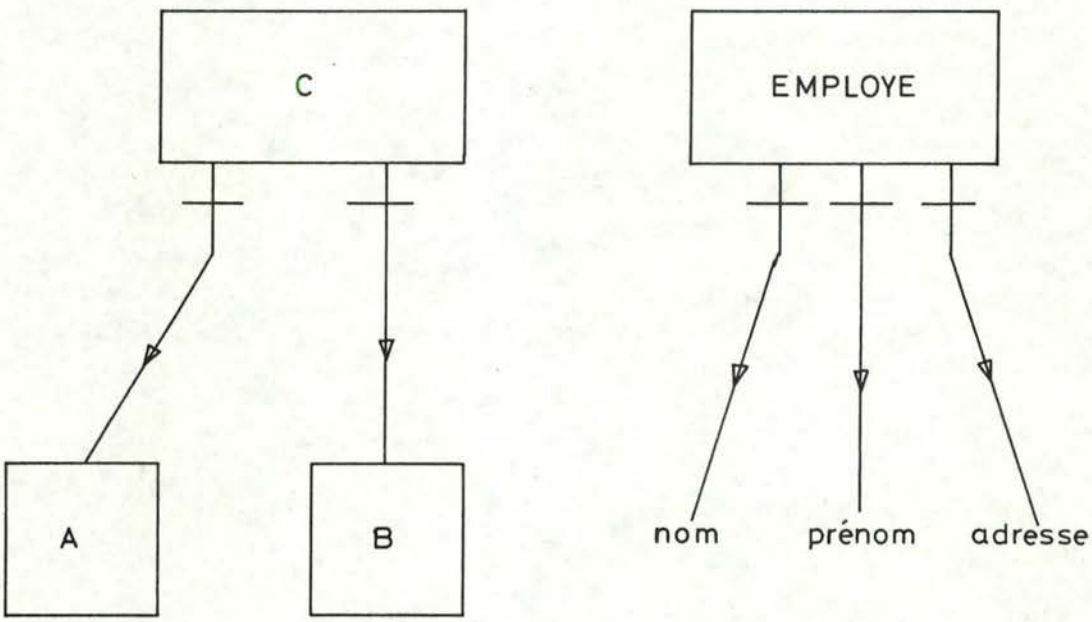
Nous exprimerons la force d'un type de chemin par un trait perpendiculaire à celui exprimant le type de chemin.
Ce trait sera placé du côté du ou des types d'articles ou des items pour lesquels le type de chemin est fort.

Exemple



La cardinalité d'un type de chemin est une expression de la capacité d'un type d'article à en identifier un autre. Il est possible que deux ou plusieurs articles de types différents (A , B par exemple) n'identifient pas un article d'un autre type (C par exemple) mais que leur combinaison l'identifie (un article a de A et un article b de B identifient un article c de C). Nous pouvons représenter cette propriété en utilisant la convention graphique décrite ci-dessous. Cette représentation est également applicable aux listes d'items constituant un identifiant d'un type d'article.

Exemples :



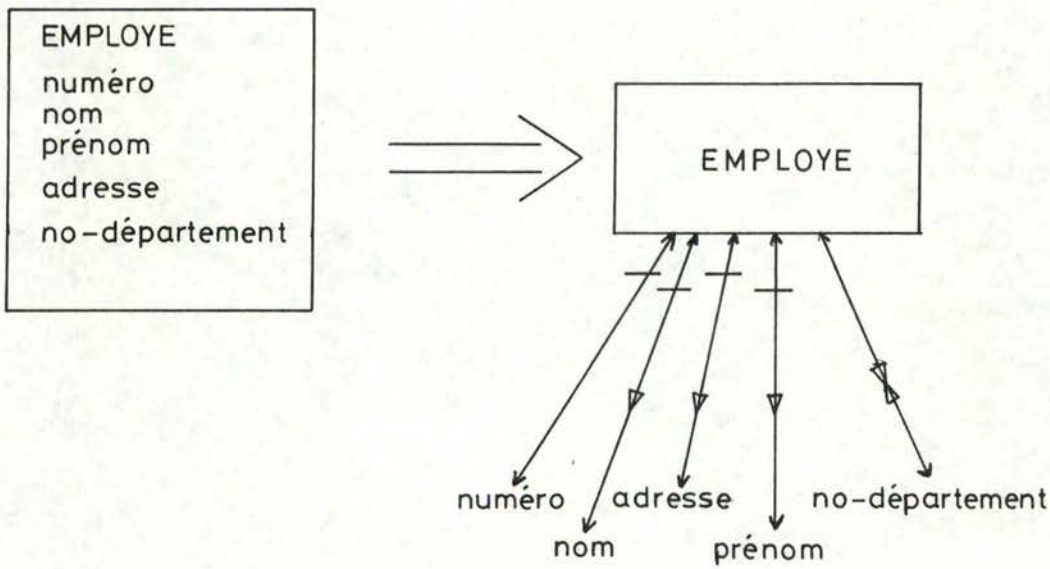
D. Transposition du modèle d'analyse conceptuelle en un modèle d'accès.

Nous allons ici présenter les règles de passage du modèle conceptuel au modèle d'accès. Nous trouverons un exposé complet des règles de transition dans [8].

1. Représentation d'un type d'entités et de ses propriétés

A chaque type d'entité, on associe un type d'article tel que toute entité du premier type est représenté par un article du second type. Chaque propriété d'un type d'entité est représentée par un item associé au type d'articles relatif à ce type d'entité. Chaque valeur de l'un étant représentée par une valeur de l'autre. Les propriétés de l'item (type de valeur, facultatif/obligatoire, élémentaire/décomposable, simple/répétitif, identifiant/non identifiant) sont en relation avec le type de la propriété correspondante.

Exemple :



numéro : identifiant, obligatoire
nom : non " "
adresse : " "
prénom : " "

no-département : " facultatif

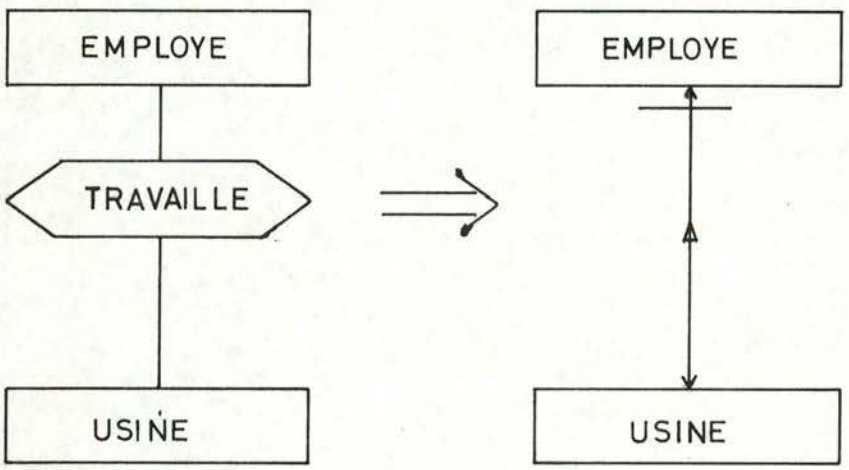
2. Représentation d'un type de relation binaire sans propriété

Soient A et B deux types d'entité du schéma conceptuel représentés par les types d'article AA et AB. Soit R (A,B) type de relation entre A et B sans propriété.
R sera représenté par un type de chemin, noté ici CHR, d'origine AA et de cible AB, de même connectivité et existence que R et

tel que si a appartient à A
b appartient à B
(a,b) appartient à R alors b' est cible dans
a' représente a le chemin CHR
b' représente b d'origine a'.

On construit également le type de chemin inverse de CHR.

Exemple :



TRAVAILLE (EMPLOYE, USINE)
CARDINALITE : (1 - N)

3. Représentation d'un type de relations avec propriété.

Nous utiliserons pour cela la création d'un type d'article dans le modèle d'accès qui ne représente pas un type d'article mais bien un type de relations.

Considérons un type de relation R défini sur A, B et doté des propriétés X et Y.
La représentation sera construite, dans l'ordre, de la manière suivante :

1. On définit un nouveau type d'article, noté AR tel que chaque article de ce type représente une occurrence de la

relation R et que toute occurrence de R soit ainsi représentée.

2. Si R est doté de propriétés, chacune d'elles est représenté par un item associé au type d'article AR. Les propriétés de ces items sont choisies de manière à représenter le type des propriétés correspondantes.

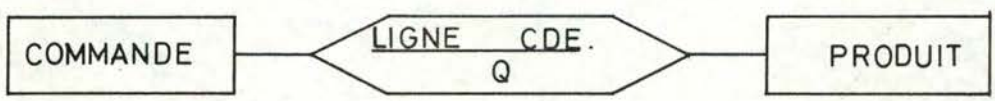
3. On représente la participation à R de chacun de ses membres (ici A et B) par un type de chemin d'accès dont l'origine est le type d'article représentant ce membre et la cible, le type d'article AR. Appelons-les respectivement CHA, CHB.

4.La connectivité et l'existence de ces types de chemin est définie comme suit :
pour l'origine, on conserve la connectivité au membre du type de relation.
pour la cible, on impose que le type de chemin soit fort.

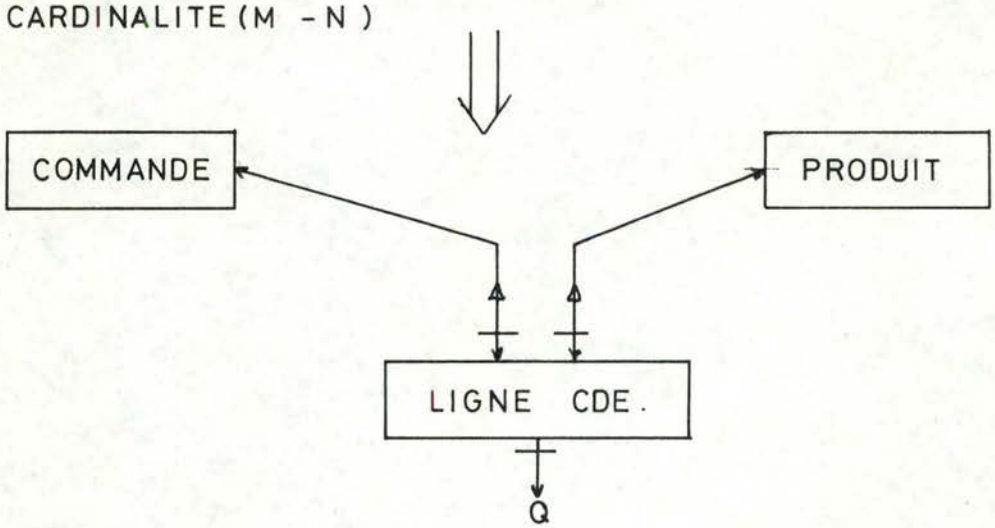
5. Pour chaque identifiant du type de relations, on impose au nouveau type d'article AR un groupe identifiant; les composants de ce dernier sont les types de chemin construits en 3 dont l'origine représente un type d'entité de l'identifiant.

6. A chacun des types de chemin construits en trois, on associe son inverse. Si R' est une occurrence de la relation R = (a, b), alors l'article cible du chemin inverse de CHA d'origine R' représente l'entité a.

Exemples



LIGNE CDE. (COMMANDE, PRODUIT)
CARDINALITE (M - N)



E. Langage de descritption des applications

Nous avons retenu un langage simple tirè de deux langages connus Algol et Pascal. Nous y retrouvons les principales structures nècessaires à l'expression d'un programme. Nous allons les dècrire brièvement. Les mots soulignès sont des mots rèservès du langage. On trouvera un exposè plus complet de ce langage dans [15].

a) Structures algorithmiques

1) affectation : x: = E

x prend la valeur rèsultat de l'expression E

2) conditionnelles : IF B THEN S1 ELSE S2 FI

Si B est vrai alors S1 est effectuè
Si B est faux alors S2 est effectuè

(B reprèsente toute expression pouvant être èvaluèe comme vraie ou fausse)

3) itèration : While B do S od

Tant que B sera vrai, S sera effectuè

(B reprèsente toute expression poouvant être èvaluèe)

4) procèdure : NOM (arguments , rèponses)

On exprimera pour celles-ci ,

ses entrèes : arguments
ses sorties : rèponses.

Exemple : CALCULER SALAIRE

(arguments : nom-employè, index mois ...,
rèponses : salaire,...)

b) Structures de données

C.N. Tout accès doit être complètement défini.

1) Opération d'accès à la base

1.1 accès simple

a) FOR identif. = article-cible (key =v)

b) FOR identif. = article-cible FROM article-origine
VIA chemin
DO
traitement
OD;

- On accède à un article qui satisfait à la valeur de clé d'accès définie par (key = v) (a).

- On accède à un article cible du chemin déclaré par la clause VIA et dont l'article origine est défini par la clause FROM. (b)

- L'identif. est une abréviation de l'article cible qui peut servir dans les traitements ultérieurs.

1.2 accès répétitif

a) FOR EACH identif. = article-cible (key = v)

b) FOR EACH identif. = article-cible FROM article-origine
VIA chemin
DO
traitement
OD;

- On accède à tous les articles satisfaisant à la valeur de clé d'accès définie par (key = v) (a).

- On accède à tous les articles cibles du chemin déclaré par la clause VIA et dont l'article origine est défini par la clause FROM. (b)

- Si la clause d'accès n'est pas spécifiée , on accèdera à tous les articles du type de l'article cible.

1.3 accès commandé par une condition

a) FOR [EACH] identif. = article-cible (key = v)

SUCH THAT condition

b) FOR [EACH] identif. = article-cible FROM article-origine

VIA chemin

SUCH THAT condition

DO

traitement

OD;

On accède à l'(aux) article(s) qui en plus des conditions exprimées par 1.1 et 1.2 satisfont à la condition exprimée par la clause SUCH THAT.

1.4 accès répétitif conditionné

a) FOR [EACH] identif. = article-cible (key =v)

WHILE condition

b) FOR [EACH] identif. = article-cible FROM article-origine

VIA chemin

WHILE condition

DO

traitement

OD;

On accède à l'(aux) article(s) qui en plus des conditions exprimées par 1.1, 1.2, s'il(s) existe(nt) et ce jusqu'à ce que la condition qui suit la clause WHILE ne soit plus respectée.

2) opération de modification de l'état de la base de donnée

- CREER
- SUPPRIMER
- METTRE A JOUR

Exemple :

CREER EMPLOYE (nom-employé = DUPONT, Prénom employé = JEAN ...)

c) Structure de synchronisation

ATTENDRE (condition)
SIGNALER (événement)

d) Structure de communication

ENVOYER ‹nom de donnée› à ‹NOM-PROGR.›

RECEVOIR " de <NOM-PROGR.>

e) Entrées - Sorties

INTRODUIRE < nom de donnée>
SORTIR < nom de donnée>

N.B. Pour référencer les items d'un article , nous utiliserons la notation suivante : nom-employé (EMPLOYE)

DEUXIEME PARTIE

Deuxième partie : Analyse conceptuelle du système

A. Introduction.

Dans cette deuxième partie, nous allons présenter une brève description du système CORRESPONDANTS. De celle-ci, nous tirerons le schéma conceptuel du système. Nous décrirons ensuite, en utilisant les formalismes du modèle d'analyse conceptuelle, les entités, les relations et les attributs identifiés par le schéma. Nous exprimerons les contraintes autres que celles de cardinalité et d'existence, et nous analyserons les applications du système. Tout ceci constituera la base des analyses ultérieures.

B. Description sommaire du Système CORRESPONDANTS.

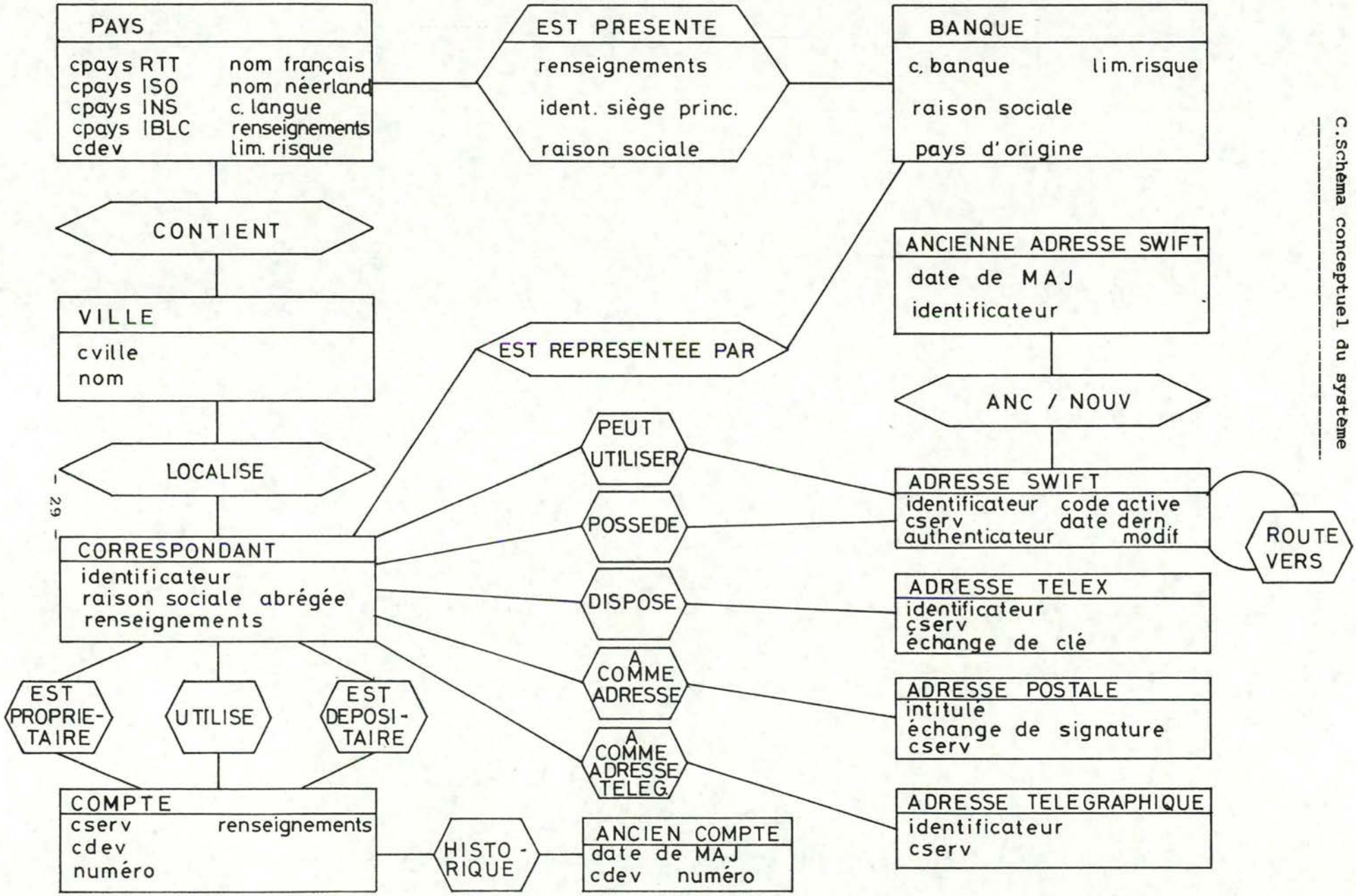
Le but de ce système est de rassembler l'ensemble des informations sur les correspondants banquiers de la banque. Ces informations se trouvaient dans différents fichiers et, certaines applications devant travailler sur ces données, passaient continuellement d'un fichier à l'autre. Il faut aussi savoir que le système que nous allons étudier est appelé à se développer. Il conviendra donc d'en tenir compte dans notre analyse. Dans cette description, nous allons présenter brièvement les données et les traitements concernés par ce système. Nous ne pouvons reproduire l'ensemble des documents et conversations ayant servis à établir l'analyse conceptuelle. Ils ne seraient d'ailleurs que de peu d'intérêt pour le lecteur.

Dans ce système, nous considérons des pays identifiés par divers codes (expl. code IBLC). Nous connaissons un certain nombre d'informations sur les pays telles leur nom, leur devise, les langues que l'on y parle et divers renseignements transmis par la ligne internationale (service traitant les relations avec l'extérieur) sur les opérations admissibles ou non avec ces pays. Dans ces pays se trouvent des banques caractérisées par un code identifiant, une raison sociale, un siège principal et autres renseignements produits par la ligne internationale. Un correspondant est le siège d'une banque dans une ville. Il possède éventuellement un certain nombre d'adresses sur le réseau inter-bancaire SWIFT. On connaît également leur(s) adresse(s) postale(s), leur(s) adresse(s) télex et télégraphique(s), de même que les différents comptes qui leur appartiennent, comptes qui peuvent être bien sûr déposés ailleurs que chez eux. Nous devons également savoir quels comptes peut utiliser un correspondant autre

que ceux lui appartenant.

Diverses caractéristiques sont également connues sur les adresses SWIFT et Télex, tel le fait qu'il y ait ou non échange de clés de contrôle, que l'adresse SWIFT soit active ou non... Au sujet de l'adresse SWIFT, nous devons envisager le fait qu'elle devienne inaccessible. Dans ce cas, une adresse de remplacement est prévue. Enfin, une adresse SWIFT et un compte peuvent disparaître et être remplacés par plusieurs. Il faut garder une trace de tels changements pour les statistiques annuelles. Nous ne donnons pas ici la quantification des informations, nous la trouverons à la fin de l'analyse conceptuelle. De même, nous ne décrirons pas ici précisément les applications du système. On pourrait résumer l'essentiel de celles-ci de cette manière : elles servent au bon acheminement des messages de et vers l'étranger pour tous types d'adressage, et à donner les comptes à mouvementer en fonction de l'adresse SWIFT ou de l'identificateur du correspondant. D'autres applications produisent la liste des correspondants soit par PAYS / VILLE , soit par BANQUE / PAYS / VILLE . Sur ces listes, on trouvera également les numéros de compte des correspondants.

Dans l'analyse conceptuelle, nous n'avons pas analysé les mises à jour. C'est en effet une procédure tout à fait particulière qui a lieu tous les deux mois. Ces mises à jour proviennent de la ligne internationale. Elles sont relativement peu nombreuses et assez différentes d'une fois à l'autre. Nous considérerons ici que nous devons pouvoir mettre à jour toutes les entités et propriétés énoncées dans le schéma.


PAYS
cpays RTT
cpays ISO
cpays INS
cpays IBLC
cdev
nom français
nom néerland
c. langue
renseignements
lim. risque
EST PRESENTE
renseignements
ident. siège princ.
raison sociale
BANQUE
c. banque
lim. risque
raison sociale
pays d'origine
CONTIENT
VILLE
cville
nom
EST REPRESENTEE PAR
ANCIENNE ADRESSE SWIFT
date de MAJ
identificateur
ANC / NOUV
LOCALISE
PEUT UTILISER
POSSEDE
DISPOSE
A COMME ADRESSE
A COMME ADRESSE TELEG.
ADRESSE SWIFT
identificateur
cserv
authenticateur
code active
date dern. modif
ROUTE VERS
ADRESSE TELEX
identificateur
cserv
échange de clé
ADRESSE POSTALE
intitulé
échange de signature
cserv
ADRESSE TELEGRAPHIQUE
identificateur
cserv
CORRESPONDANT
identificateur
raison sociale abrégée
renseignements
EST PROPRIE-TAIRE
UTILISE
EST DEPOSI-TAIRE
COMPTE
cserv
cdev
numéro
renseignements
HISTO-RIQUE
ANCIEN COMPTE
date de MAJ
cdev
numéro

D. Expression sémantique des entités.

PAYS

{ si p appartient à PAYS alors p désigne un pays de la terre connu de la banque propriétaire du système d'information :

dont les langues que l'on y parle sont représentées par
(p.clangue) [ex. clangue = 1 => français]

dont la devise principale est représntée par
(p.cdev) [ex. cdev = 1 => FB]

dont la dénomination française est représentée par
(p.nom français) [ex. BELGIQUE]

dont la dénomination néerlandaise est représentée par
(p.nom néerlandais) [ex. BELGIE]

dont le code pays RTT non identifiant est représenté par
(p.cpays RTT) [ex. cpays RTT = 3]

dont le code pays INS non identifiant est représenté par
(p.cpays INS) [ex. cpays INS = 2]

dont le code pays ISO non identifiant est représenté par
(p.cpays ISO) [ex. cpays ISO = 5]

dont le code pays IBLC identifiant est représenté par
(p.cpays IBLC) [ex. cpays IBLC = 4]

dont les renseignements qui représentent ce que la ligne internationale nous a transmis sont représentés par
(p.renseignements)
[ex. ne traiter aucune opération comportant un risque avec les banques de ce pays sans en référer à la direction de la ligne internationale.]

dont le montant maximum des risques que l'on peut prendre sur ce pays est représenté par
(p.limite risque)

BANQUE

{ si b appartient à BANQUE alors b désigne un organisme bancaire indépendamment de tout pays
dont le code identifiant est représenté par
(b.cbanque) [ex. cbanque = 3]
dont la dénomination est représentée par
(b.raison sociale) [ex. BANCO DI ROMA]
dont le pays d'origine est représenté par
(b.pays d'origine) [ex. BBL est d'origine belge : BELGIQUE]
dont le montant maximum des risques que l'on peut prendre sur ce pays est représenté par
(b.limite résque)}

VILLE

{ si v appartient à VILLE alors v désigne une ville appartenant à un pays
dont le code identifiant est représenté par
(v.cville) [ex. cville = MCH]
dont la dénomination est représentée par
(v.nom) [ex. v.nom = MUNICH]

COMPTE

{ si c appartient à COMPTE alors c représente un compte bancaire que la banque propriétaire de la base peut soit créditer, soit débiter
dont le numéro est représenté par
(c.num) [ex. 301.0175949.72]
dont la devise est représentée par
(c.dev) [ex. c.dev = 75]
dont les services du correspondant propriétaire pouvant utiliser le compte sont représentés par
(c.cserv)
Le code service donne les différents services habilités à utiliser l'objet auquel il se rapporte :
ex. service change valeur du code 00001
service arbitrage 00100
service change + arbitrage 00101
dont les renseignements que l'on connaît sont représentés par
(c.renseignements)
[ex. compte 3 : pas de compte financier; faire double arbitrage et créditer en compte convertible]. }

CORRESPONDANT

{ si c appartient à CORRESPONDANT alors c désigne le siège d'une banque b appartenent à BANQUE dans une ville v appartenent à VILLE

dont l'identificateur est la concaténation du code identifiant de la banque à laquelle appartient le correspondant, du c.pays identifiant de son pays et du code de la ville où il est situé

[ex. BOFA US MCH]

dont la raison sociale est représentée par

(c.rais.sociale abrégée) [ex. BNPA]

dont les renseignements que l'on connaît sont représentés par

(c.renseignements) [ex. banque à favoriser] }

ADRESSE TELEX

{ si at appartient à ADRESSE TELEX alors at désigne une adresse télex appartenant à un des correspondants de la base

dont l'identificateur est le numéro d'appel de l'appareil physique et est désigné par

(at.identificateur)

dont le fait qu'il y ait échange de clé télégraphique entre son propriétaire et la banque propriétaire du système d'information est représenté par

(at.échange de clé) [ex. échange de clé Y]

dont les différents services auxquels elle est accessible sont représentés par

(at.cserv)

ADRESSE SWIFT

{ si as appartient à ADRESSE SWIFT alors as désigne une adresse sur le réseau de communication inter-banques SWIFT

dont l'identificateur formé du code banque, code pays, code de localisation et du code de branchement est représenté par

(as.identificateur) code banque :

code qui indique la banque du propriétaire de l'adresse SWIFT

code pays :

code qui indique le pays du propriétaire de l'adresse SWIFT

code de localisation :

code qui indique la ville du propriétaire de l'adresse SWIFT

code de branchement :

code purement interne au propriétaire qui peut par exemple grâce à lui, diriger les communications qui arrivent aux services concernés

dont le fait que les messages reçus doivent être accompagnés ou non de clés de contrôle est représenté par

(as.authenticateur) [ex. authenticateur = Y]

dont le fait qu'elle soit active ou non est représenté par
(as.active) [ex. as.active = Y]
dont la date de dernière modification est
(as.date dern. modif.) [ex. 790122]
dont les différents services auxquels elle est accessible sont représentés par
(as.cserv)

ANCIENNE ADRESSE SWIFT

{ si as appartient à ANCIENNE ADRESSE SWIFT alors as est une ancienne adresse sur le réseau qui a été remplacée à la date représentée par
(as.date de maj)
dont l'identificateur formé du code banque, du code pays, du code localisation et du code de branchement est représenté par
(as.identificateur) }

ANCIEN COMPTE

{ si cpt appartient à ANCIEN COMPTE alors cpt est un ancien compte qui a été remplacé à la date représentée par
(cpt.date de maj)
dont le numéro est représenté par
(cpt.numéro)
dont le code de la devise dans laquelle il est libellé est représenté par
(cpt.cdev) }.

ADRESSE POSTALE

{ si ap appartient à ADRESSE POSTALE alors ap est une adresse postale
dont le libellé est représenté par
(ap.intitulé)
dont le fait qu'il y ait échange de signature entre la banque propriétaire du système et le propriétaire de l'adresse permet l'envoi de certains types de documents, est représenté par
(ap.échange de signature)
dont les services qui l'utilisent sont représentés par
(ap.cserv) }.

ADRESSE TELEGRAPHIQUE

{ si at appartient à ADRESSE TELEGRAPHIQUE alors at est une adresse télégraphique
dont l'identificateur est la représentation de cette adresse
(at.identificateur)
dont les services qui l'utilisent sont représentés par
(at.cserv) }.

E. Expression sémantique et contraintes de cardinalité et d'existence des relations du schéma.

* CONTIENT (PAYS , VILLE) cardinalité 1-N

{ (p,v) : la ville (v) est située dans le pays (p) }.

* LOCALISE (VILLE , CORRESPONDANT) cardinalité 1-N

{ (v,c) : le correspondant (c) est un siège bancaire situé dans la ville (v) }.

* EST PRESENTE (BANQUE , PAYS) cardinalité M-N

{ (b,p) : la banque (b) exerce une présence dans le pays (p)
dont les renseignements transmis par la ligne internationale sont représentés par
(bp.renseignements)
dont on peut ientifier le siège principal par
(bp.ident.siège princ)
dont la raison sociale est représentée par
(bp.raison sociale) }.

* PROPRIETAIRE (CORRESPONDANT , COMPTE) cardinalité 1-N

{ (c,cpt) : le correspondant (c) est propriétaire du compte (cpt) }.

* EST DEPOSITAIRE (CORRESPONDANT , COMPTE) cardinalité 1-N

{ (c.cpt) : le compte (cpt) est déposé chez le correspondant (c) }.

* UTILISE (CORRESPONDANT , COMPTE) cardinalité M-N

{ (c,cpt) : le correspondant (c) peut utiliser le compte (cpt) dont il n'est pas nécessairement propriétaire }.

* POSSEDE (CORRESPONDANT , ADRESSE SWIFT) cardinalité 1-N

{ (c,as) : le correspondant (c) a comme adresse sur le réseau SWIFT (as) }.

* ROUTE VERS (ADRESSE SWIFT , ADRESSE SWIFT) cardinalité M-N

{ (as1,as2) : l'adresse (as2) remplace l'adresse (as1) si celle-ci devient inutilisable}

* DISPOSE DE (CORRESPONDANT , ADRESSE TELEX)

cardinalité 1-N

{ (c,at) : le correspondant (c) a comme adresse Télex (at) }.

* ANC/NOUV (ADRESSE SWIFT , ANCIENNE ADRESSE SWIFT)

cardinalité M-N

{ (as1,as2) : l'adresse (as1) remplace l'adresse (as2) sur le réseau SWIFT }.

* HISTORIQUE (COMPTE , ANCIEN COMPTE) cardinalité M-N

{ (c1,c2) : le compte (c1) remplace le compte (c2) }.

* UTILISE (CORRESPONDANT , ADRESSE SWIFT) cardinalité M-N

{ (c,as) : le correspondant (c) utilise l'adresse SWIFT as }.

* A COMME ADRESSE (CORRESPONDANT , ADRESSE POSTALE)

cardinalité 1-N

{ (c,ap) : le correspondant (c) a comme adresse postale (ap) }.

* A COMME ADRESSE TELEGRAPHIQUE
(CORRESPONDANT , ADRESSE TELEGRAPHIQUE) cardinalité 1-N

{ (c,at) : le correspondant (c) a comme adresse télégraphique (at) }.

F. Classification des propriétés

ENTITE PAYS

cpays RTT	non identifiant	facultatif	
cpays ISO	"	"	
cpays INS	"	"	
nom français	"	"	
nom néerlandais	"	"	
cdev	"	"	
renseignements	"	"	
clangue	"	"	répétitif
cpays IBLC	"	"	
limite risque	"	"	

ENTITE VILLE

cville	identifiant,	obligatoire
nom	non identifiant,	facultatif

ENTITE BANQUE

cbanque	identifiant	, obligatoire	
raison sociale	non identifiant	, facultatif	
pays origine	"	"	
limite risque	"	"	

ENTITE CORRESPONDANT

identificateur	identifiant	, obligatoire	
raison sociale abrégée	non identifiant	, facultatif	
renseignements	"	"	

ENTITE ADRESSE SWIFT

identificateur	identifiant	, obligatoire	
cserv	non identifiant	, obligatoire	
authenticateur	"	"	
code active	"	"	
date dern. modif	"	"	

ENTITE ADRESSE TELEX

identificateur	identifiant	obligatoire
cserv	non identifiant	obligatoire
échange de clé	"	facultatif

ENTITE COMPTE

cserv	non identifiant	obligatoire
numéro	identifiant	"
cdev		
renseignements	non identifiant	facultatif

RELATION EST PRESENTE

renseignements	non identifiant	facultatif
ident. siège princ.	identifiant	facultatif
raison sociale	non identifiant	"

ENTITE ANCIEN COMPTE

date de maj	non identifiant	obligatoire
numéro	identifiant	obligatoire
cedv		

ENTITE ANCIENNE ADRESSE SWIFT

date de maj	non identifiant	obligatoire
identificateur	identifiant	obligatoire

ENTITE ADRESSE POSTALE

intitulé	identifiant	obligatoire
échange signature	non identifiant	facultatif
cserv	non identifiant	obligatoire

ENTITE ADRESSE TELEGRAPHIQUE

identificateur	identifiant , obligatoire
cserv	non identifiant , obligaotire

G. Enoncé des contraintes.

C 1.

L'ensemble des comptes que les correspondants peuvent utiliser doit être inclus dans l'ensemble des comptes dont les correspondants sont propriétaires.

C 2.

L'ensemble des comptes qui sont déposés chez les correspondants doit être inclus dans l'ensemble des comptes dont les correspondants sont propriétaires.

Ces deux contraintes se vérifient grâce à la relation forte PROPRIETAIRE (CORRESPONDANT , COMPTE) qui impose que tout compte ait un propriétaire. Dans ce cas l'ensemble des comptes dont les correspondants sont propriétaires est un ensemble maximum et donc les deux autres ensembles sont inclus dans celui-ci.

C 3.

Il existe un et un seul correspondant par banque et pays dans une ville.

C 4.

Si un correspondant (c) exerce son activité dans une ville (v) d'un pays (p), alors la banque (b) représentée par (c) doit être présente dans le pays (p) qui contient la ville (v).

C 5.

Par correspondant, il n'y a qu'une adresse swift pour une valeur déterminée du code service.

C 6.

Par correspondant, il n'y a qu'un compte pour une valeur déterminée du code service.

C 7.

Par correspondant, il n'y a qu'une adresse postale pour une valeur déterminée du code service.

C 8.

Par correspondant, il n'y a qu'une adresse télégraphique pour une valeur déterminée du code service.

C 9.

Par correspondant, il n'y a qu'une adresse telex pour une valeur déterminée de code service.

H. Description des applications

APPLICATION 1 : RECHERCHE DE L'ADRESSE SWIFT/TELEX SUR BASE D'UN COMPTE (1600 * par jour)

ENTREES

numéro de compte : NC
code devise : CD
code service : CS

SORTIES

adresse SWIFT
adresse TELEX
code résultat

FONCTION

Cette application fournit, si elle existe, l'adresse SWIFT dont est propriétaire, ou que peut utiliser, le propriétaire du compte (NC,CD). Cette adresse doit être utilisable par le(s) service(s) dont le code est (CS). A défaut, ou si l'adresse SWIFT n'est pas active, l'application fournira l'adresse TELEX, si elle existe, du propriétaire du compte pour le service demandé. Dans tous les cas, l'application précisera le code résultat de la recherche avec ces valeurs suivantes :

0 = pas d'adresse
1 = adresse SWIFT du propriétaire
2 = adresse SWIFT utilisée
3 = adresse TELEX
4 = numéro de compte - devise inexistant.

APPLICATION 2 : RECHERCHE D'ADRESSE TELEGRAPHIQUE / POSTALE SUR BASE D'UN IDENTIFICATEUR ET D'UN CODE SERVICE (50 * par jour)

ENTREES

identificateur : ID
code service : CS

SORTIES

adresse télégraphique
adresse postale
code résultat

FONCTION

Cette application fournit, si elles existent, les adresses télégraphiques et postales pour le code service (CS) du correspondant dont l'identificateur = (ID). Elle positionne le code résultat avec les valeurs suivantes :

0 = pas d'adresses
1 = adresse télégraphique
2 = adresse postale
3 = deux adresses
4 = identificateur inexistant

APPLICATION 3 : RECHERCHE DE L'ADRESSE TELEX SUR BASE DE L'IDENTIFICATEUR / DU CODE SERVICE (50 * par jour)

ENTREES

identificateur : ID
code service : CS

SORTIES

adresse TELEX
code résultat

FONCTION

Cette application fournit, si elle existe, l'adresse TELEX pour le code service (CS) dont est propriétaire le correspondant identifié par (ID). Le code résultat de la recherche est positionné avec les valeurs suivantes :

0 = pas d'adresse
1 = adresse TELEX
2 = identificateur erroné

APPLICATION 4 : RECHERCHE D'UN NOM DE BANQUE/PAYS SUR BASE D'UN COMPTE (50 * par jour)

ENTREES

numéro de compte : NC
code devise : CD

SORTIES

nom de banque/pays - ville du correspondant
code résultat

FONCTION

Cette application fournit le nom de la banque / pays (présence d'une banque dans un pays) dont dépend le correspondant propriétaire du compte (NC,CD). Le code résultat de la recherche peut prendre les valeurs suivantes :

0 = compte inexistant
1 = nom de banque / pays connu
2 = nom de banque / pays inconnu

APPLICATION 5 : RECHERCHE D'UN CODE PAYS IBLC SUR BASE DU CODE PAYS INS (10 * par jour)

ENTREE

code pays INS : CINS

SORTIES

code pays IBLC
code résultat

FONCTION

Cette application fournit les codes pays IBLC des pays dont le code INS est CINS. Le code résultat de la recherche prendra les valeurs suivantes :

0 = CINS est inexistant
i = 1,..n il y a i pays qui possèdent ce code et il y aura i code IBLC en sortie.

APPLICATION 6 : RECHERCHE DES NOMS DE PAYS SUR BASE DE LEUR CODE IBLC (10 * par jour)

ENTREE

code pays IBLC : CIBLC

SORTIES

nom du pays en français
nom du pays en néerlandais
code résultat

FONCTION

Cette application fournit le nom en français et en néerlandais du pays dont le code IBLC est CIBLC. Le code résultat de la recherche prendra les valeurs suivantes :

0 = code IBLC inexistant
1 = les noms du pays sont connus
2 = les noms du pays sont inconnus

APPLICATION 7 : RECHERCHE D'UN COMPTE SUR BASE D'UNE ADRESSE SWIFT/ D'UN CODE SERVICE/ D'UN CODE DEVISE. (1 600 * par jour)

ENTREE

adresse SWIFT : ID
code devise : CD
code service : CS

SORTIES

numéro de compte
code devise
identificateur du propriétaire du compte
identificateur du dépositaire du compte
code résultat

FONCTION

Cette application fournit sur base de l'adresse SWIFT (ID) d'un correspondant et de la devise (CD) le numéro du compte valable pour les services représentés par cserv, s'il existe. Soit le compte appartient au correspondant dont l'identificateur = (ID), soit c'est un compte qu'il peut utiliser. Dans les deux cas, l'application fournira l'identificateur du propriétaire et du dépositaire du compte retenu. Le code résultat sera positionné de cette façon :

0 = identificateur inexistant
1 = pas de compte utilisable
2 = compte du correspondant entré
3 = compte d'un autre correspondant

APPLICATION 8 : LISTE DES CORRESPONDANTS PAR PAYS / VILLE (1 * tous les six mois)

ENTREE

SORTIES

- noms de pays - renseignements
- nom de ville
- raison sociale des banques/pays des correspondants
- numèros de compte des correspondants - renseignements
- code devise

FONCTION

Cette application produit la liste des correspondants avec leurs numèros de compte et renseignements sur ceux-ci. Cette liste est produite par pays dont le nom ainsi que les renseignements sont sortis, et par ville dont le nom est sorti.

APPLICATION 9 : LISTE DES CORRESPONDANTS PAR BANQUE (1 * tous les six mois)

ENTREE

SORTIES

- raison sociale des banques - nom de pays
- raison sociale des banques / pays - renseignements
- nom des villes
- numèros de comptes - renseignements

FONCTION

Cette application produit la liste des correspondants avec leurs numèros de compte et renseignements sur ceux-ci. Cette liste est produite par banque dont la raison sociale est sortie, par banque / pays dont la raison sociale et les renseignements sont sortis, par ville dont le nom est sorti.

I. Quantifications

Il s'agit ici de déterminer les éléments de base qui serviront à l'évaluation des applications. On déterminera le nombre d'entités de chaque type et le nombre moyen d'entités cibles d'une relation pour une entité origine.

Type d'entité PAYS

Tous les pays de la terre sont repris dans ce système. On en dénombre environ : 200

Type d'entité BANQUE

On entretient des relations avec environ 2 000 banques.

Type d'entité CORRESPONDANT

On connaît des informations sur environ 6 000 correspondants

Type d'entité COMPTE

5 200 comptes sont dénombrés.

Type d'entité ADRESSE TELEGRAPHIQUE

environ 7 000 adresses télégraphiques sont connues.

Type d'entité ADRESSE POSTALE

environ 9 000

Type d'entité VILLE

Les correspondants que nous connaissons se situent dans 1 000 villes.

Type d'entité ADRESSE SWIFT

environ 3 800 adresses swift nous sont connues

Type d'entité ADRESSE TELEX

environ 7 000

Type d'entité ANCIENNE ADRESSE SWIFT

Relativement peu

Type d'entité ANCIEN COMPTE

relativement peu

Nombre moyen de villes par pays

1 000 : 200 = 5

Nombre moyen de correspondants par ville

6 000 : 6 - 6

Nombre moyen d'adresses swift par correspondant

3 800 : 600 = 0.6

Nombre moyen d'adresses télex par correspondant

7 000 : 6 000 = 1.2

Nombre moyen d'adresses télégraphiques par correspondant

7 000 : 6 000 = 1.2

Nombre moyen d'adresses postales par correspondant

9 000 : 6 000 = 1.5

Nombre moyen de comptes appartenant à un correspondant

5 200 : 6 000 = 0.87

Les autres quantifications seront définies dans la description de chaque application.

TROISIEME PARTIE

Troisième Partie : Etude des accès

A. Introduction

Dans cette troisième partie, nous allons d'abord réaliser la transformation du schéma conceptuel en un schéma des accès logiques.. Pour cette première proposition, nous retiendrons tous les accès possibles. Dans un deuxième temps nous réaliserons l'analyse des applications en terme d'accès à la base. Pour réaliser celle-ci, nous effectuerons les différentes étapes décrites ci-dessous :

1.- Description de l'algorithme.

2.- Rappel des éléments quantitatifs nécessaires à l'analyse et au calcul des grandeurs spécifiques.

3.- Analyse de l'algorithme du point de vue du nombre d'opérations effectuées.

4.- Relevé des primitives (acc's et mise à jour) nécessaires à la réalisation de l'algorithme et comptabilisation du nombre de leurs activations. Cette analyse nous permettra de déterminer le schéma des accès logiques nécessaires qui servira lors de la quatrième partie de notre étude.

B. Transformation du schéma conceptuel en un schéma des accès logiques.

Partant du schéma conceptuel, nous pouvons appliquer les transformations vues au paragraphe - Transformation du modèle E/R/P - Modèle d'accès. Nous ne détaillerons pas cette transformation qui est systématique. Nous préciserons simplement le fait que nous avons pu introduire sur notre schéma les contraintes de cardinalité, d'existence et les contraintes de C3, C5, C6, C7, C8 qui étaient décrites à part.

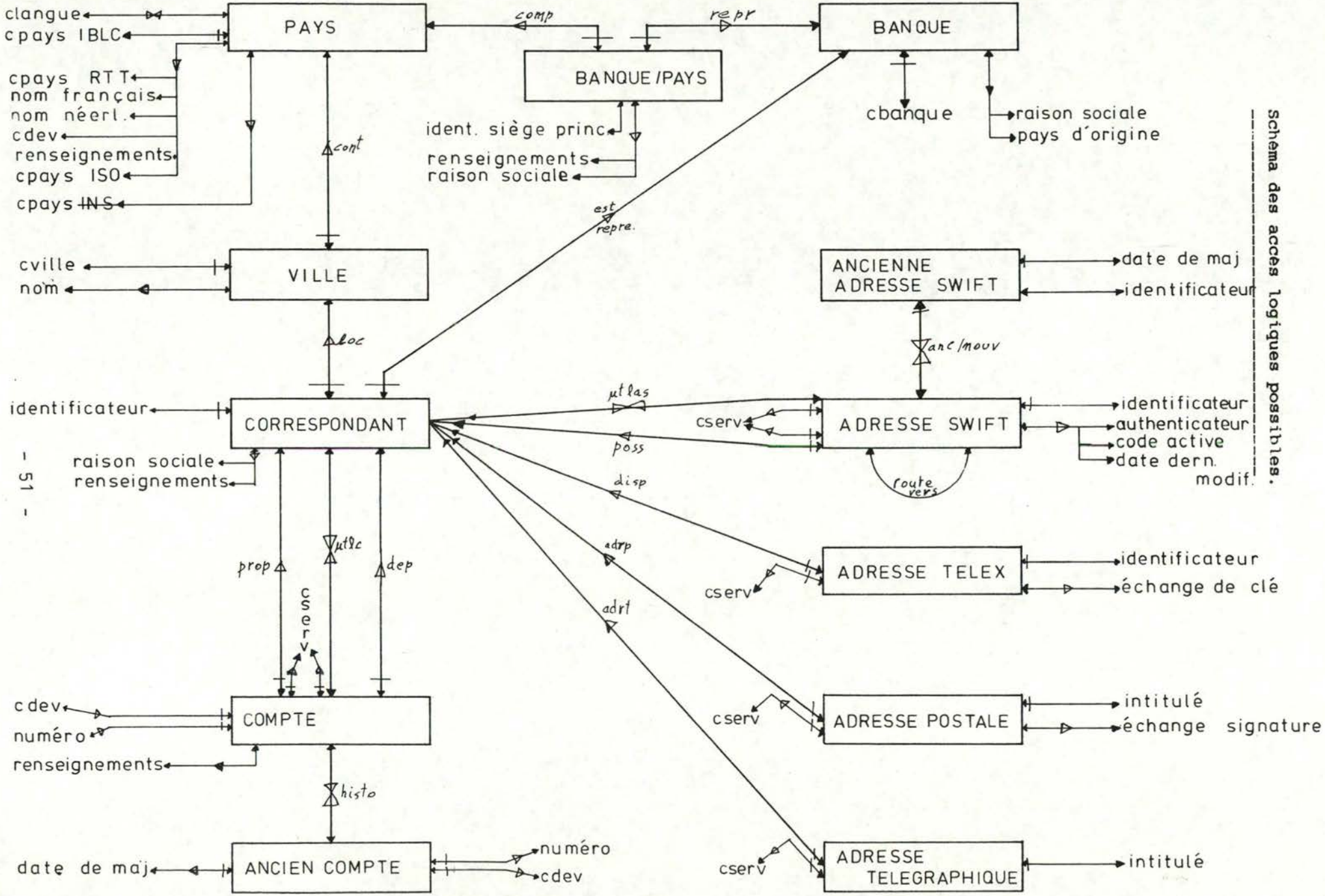


Schéma des accès logiques possibles.

C. Analyse des applications

La fonction de chaque application ainsi que ses données d'entrée et de sortie ont été décrites au cours de l'analyse fonctionnelle.

Application 1.

1.1 Description

```
Code résultat : = 4
For cpt = COMPTE (numéro = NC, cdev = CD)

    DO code résultat : = 0;
       For C = CORRESPONDANT FROM cpt VIA prop.

        DO FOR as = ADRESSE SWIFT FROM C VIA poss such that
                                              cserv (as) = cs

          DO IF code active = True
             THEN code résultat : = 1;
                 adresse swift : = Identificateur (as);
             FI
          OD
          FOR as = ADRESSE SWIFT FROM C VIA utlas such that
                                        cserv (as) = cs
                                  AND code résultat = 0
           DO IF code active = true
              THEN code résultat : = 2;
                   adresse swift : = identificateur (as);
              FI
           OD
         FOR at = ADRESSE TELEX FROM C VIA Disp such that
                                          cserv (as) = cs
                                    AND Code résultat = 0)

           DO code résultat : = 3;
              Adresse  télex : = identificateur (at);
           OD
       OD
    OD;
```

1.2 Quantification (1 600 x par jour)

Probabilité que le compte n'existe pas = 0.05
Probabilité que le correspondant propriétaire du compte soit propriétaire d'une adresse swift pour le code service demandé : 0.70
Probabilité que le correspondant propriétaire du compte puisse utiliser une adresse swift pour le code service demandé : 0.2
Probabilité que le correspondant ait une adresse télex pour le code service déterminé : 0.6

1.3 Détail des opérations

```
1600 x {0.95 x {accès compte (numéro, cdev)
          1  x {accès correspondant (prop)
        0.7  x {accès adresse swift (poss)}
        0.2  x {accès adresse swift (utlas)}
        0.06 x {accès adresse télex (disp)}
         }}}
```

1.4 Bilan des opérations

Accès COMPTE (numéro,Cdev)	1520
" CORRESPONDANT (prop)	1520
" ADRESSE SWIFT (poss)	1064
" ADRESSE SWIFT (Utlas)	304
" ADRESSE TELEX (Disp)	91

1.5 Structures d'accès mises en oeuvre :

Clé d'accès :

numéro, cdev (COMPTE)

Chemin d'accès

-prop (COMPTE --> CORRESPONDANT)
-poss (CORRESPONDANT --> ADRESSE SWIFT)
-utlas (CORRESPONDANT --> ADRESSE SWIFT)
-disp (CORRESPONDANT --> ADRESSE TELEX)
Pour les trois derniers, une clé d'accès sur cserv est suggérée.

Application 2

2.1 Description

```
code résultat : = 4 ;
FOR C = CORRESPONDANT (IDENTIFICATEUR = ID)

        DO code résultat : = 0 :
           FOR at = ADRESSE TELEGRAPHIQUE FROM C VIA adrt
                    such that cserv (at) = cs

              DO code résultat : = 1;
                 adresse télégraphique : = intitulé (at);
              OD
            FOR ap = ADRESSE POSTALE FROM C VIA adrp
                     such that cserv (ap) = cs

              DO IF code résultat = 1
                 THEN code résultat : = 3
                 ELSE code résultat : = 2;
                 adresse postale : = intitulé (ap);
                 FI
              OD
     OD
```

2.2 Quantification (50 x par jour)

Probabilité que identificateur n'appartienne pas à la base : 0.05
Probabilité qu' un correspondant ait une adresse télégraphique : 0.6
Probabilité qu' un correspondant ait une adresse postale : 0.9

2.3 Détail des opérations

```
50 x {0.95 x { Accès CORRESPONDANT (identificateur)
       0.6  x { Accès ADRESSE TELEGRAPHIQUE (adrt) }
       0.9  x { Accès ADRESSE POSTALE (adrp) }
       }}
```

2.4 Bilan des opérations

ACCES CORRESPONDANT (identificateur)	47
ACCES ADRESSE TELEGRAPHIQUE (adrt)	28
ACCES ADRESSE POSTALE (adrp)	42

2.5 Structures d'accès mises en oeuvre

Clé d'accès :

identificateur (CORRESPONDANT)

Chemin d'accès :

adrt (CORRESPONDANT —> ADRESSE TELEGRAPHIQUE)
adrp (CORRESPONDANT —> ADRESSE POSTALE)
avec clé d'accès sur cserv suggérée

Application 3

3.1 Description

```
Code résultat : = 2;
FOR C = CORRESPONDANT (identificateur = ID)

     DO code résultat : = 0;
          FOR at = ADRESSE TELEX FROM C VIA disp such that
                   cserv (at) = cs

               DO code résultat : = 1 ;
                    Adresse télex : = identificateur (at);
               OD
     OD
```

3.2 Quantification (50 x par jour)

Probabilité que le correspondant ne soit pas présent dans la base : 0.05
Probabilité qu'un correspondant ait une adresse télex : 0.6

3.3 Détail des opérations

```
50 x {0.95 x {accès correspondant (identificateur)
      0.6 x {accès à adresse télex (disp) }}}
```

3.4 Bilan des opérations

ACCES CORRESPONDANT (identificateur)	47
ACCES ADRESSE TELEX (disp)	29

3.5 Structures mises en oeuvre

Clé d'accès :
---------- identificateur (CORRESPONDANT)

Chemin d'accès :
---

disp (CORRESPONDANT --> ADRESSE TELEX)
avec clé d'accès sur cserv suggérée.

Application 4

4.1 Description

```
Code résultat : = 0 ;
For cpt = COMPTE (numéro = NC,cdev = cd)

     DO FOR C = CORRESPONDANT FROM cpt VIA prop

          DO FOR b/p = BANQUE/PAYS FROM C VIA est reprt

               DO IF raison sociale (b/p) = "blanc"
                  THEN code résultat : = 2
                  ELSE code résultat : = 1;
                  nom de banque/pays : = raison-
               sociale(b/p);
                  FI
               OD
               FOR V = VILLE FROM C VIA loc

               DO ville du correspondant : = nom (v);
               OD
          OD
     OD
```

4.2 Quantification (50 x par jour)

Probabilité qu'il n'y ait pas de compte correspondant à (numéro,cdev) 0.05

4.3 Détail des opérations

```
50 x {0.95 x {accès COMPTE (numéro,cdev)
           x {accès CORRESPONDANT (prop)
           x {accès BANQUE / PAYS (est reprt)}
           x {accès VILLE (loc) }}}}
```

4.4 Bilan des opérations

ACCES COMPTE (numéro,cdev)	47
ACCES CORRESPONDANT (prop)	47
ACCES BANQUE / PAYS (est reprt)	47
ACCES VILLE (loc)	47

4.5 Structures d'accès mises en oeuvre

Clé d'accès :

numéro, cdev (COMPTE)

Chemin d'accès :

prop (COMPTE --> CORRESPONDANT)
est reprt (CORRESPONDANT --> BANQUE/PAYS)
loc (CORRESPONDANT --> VILLE)

Application 5

5.1 Description

```
Code résultat : = 0
For each p = PAYS (cpays INS = CINS)

     DO code résultat : = code résultat +1 ;
        Code pays IBLC (code-résultat) : = cpays IBLC (p);
```

5.2 Quantification (10 x par jour)

Type d'application d'emploi assez aléatoire tout comme les trois précédentes
Dans 95 % des cas, il y a une correspondance univoque entre un code INS et un pays. Dans les 5 % de cas restants, il y a en moyenne trois pays par code --> 95 % x 1 + 5 % x 3 = 1,1 accès par cpays INS.

5.3 Détail des opérations

```
10 x {1.1 x { accès PAYS (CINS) }}
```

5.4 Bilan des opérations

Accès PAYS (cpays IBLC)	11

5.5 Structures d'accès mises en oeuvre

Clé d'accès :

cpays INS (PAYS)

Application 6

6.1 Description

```
Code résultat : = 0 ;
For p = PAYS (cpays IBLC = CIBLC)

     DO IF nom français =  "blanc" AND nom néerlandais
                                        ="blanc"
        THEN code résultat : = 2
        ELSE nom du pays en français : = nom français (p) ;
             nom du pays en néerlandais : = nom
                                        néerlandais (p);
             code résultat : = 1 ;
        FI
     OD
```

6.2 Quantification (10 x par jour)

6.3 Détail des opérations

```
{ 10 x {ACCES PAYS(cpays IBLC)}}
```

6.4 Bilan des opérations

ACCES PAYS (cpays IBLC)	10

6.5 Structures d'accès mises en oeuvre

Clé d'accès :

cpays IBLC (PAYS)

Application 7

7.1 Description

```
Code résultat : = 0;
FOR as = adresse swift (identificateur = ID)

     DO code résultat : = 1 ;
     FOR C = CORRESPONDANT FROM as VIA poss

          DO FOR cpt = COMPTE FROM C VIA prop such that code
          résultat  = 1 AND  cserv (cpt) = cs AND cdev (cpt)
          = cd

               DO numéro de compte : = numéro (cpt);
               Code résultat : = 2 ;
               identificateur du propriétaire du compte :  =
               identificateur (c) ;
                    FOR C1 = CORRESPONDANT FROM cpt VIA Dep.

                         DO identificateur du dépositaire du
                         compte : = identificateur (c1);
                         OD
                OD
               FOR cpt = COMPTE FROM C VIA  utlc  such  that
               code résultat = 1
                                   AND cserv (cpt) =  cs  AND
               cdev (cpt) = cd

                    DO NUMERO DE COMPTE : = numéro (cpt)
                    code résultat : = 3 ;
                    FOR C1 = CORRESPONDANT FROM cpt VIA prop

                         DO identificateur  du  propriétaire
                         du compte : = identificateur (c1);
                         OD
                         FOR C1 = CORRESPONDANT FROM cpt VIA
                         dep
                         DO identificateur du dépositaire du
                         compte : = identificateur (c1);
                         OD

          OD
     OD;
```

7.2 Quantification (1 600 x par jour)

-Probabilité que l'adresse swift n'existe pas dans la base : 0.05
-Les correspondants ayant des comptes en ont trois en moyenne --> 5 200 comptes : 3 = plus ou moins 1 733 correspondants ayant des comptes.
--> 1 733 : 6 000 = 0.29 = Probabilité qu'un correspondant ait des comptes .
-Probabilité que le correspondant n'ait pas de compte à lui : 0.70

7.3 Détail des opérations

```
1 600 x {0.95 x { ACCES ADRESSE SWIFT (identificateur)
          1   x { accès CORRESPONDANT (poss)
        0.3   x {Accès compte (prop)
                 1 x {ACCES CORRESPONDANT (dep) }}
        0.7   x {Accès compte (utlc)
                 1 x {Accès CORRESPONDANT (prop)}
                 1 x {Accès CORRESPONDANT (dep) }}. }}
```

7.4 Bilan des opérations

ACCES adresse swift (identif)	1 520
ACCES CORRESPONDANT (poss)	1 520
ACCES COMPTE (prop)	456
ACCES CORRESPONDANT (dep)	1 520 (456+1 064)
ACCES COMPTE (utlc)	1 064
ACCES CORRESPONDANT (prop)	1 064

7.5 Structures d'accès mises en oeuvre

Clé d'accès :

identificateur (ADRESSE SWIFT)

Chemin d'accès :

prop (CORRESPONDANT --> COMPTE)
(avec clé d'accès : cdev, cserv)
dep (COMPTE --> CORRESPONDANT)
utlc (CORRESPONDANT --> COMPTE)
(avec clé d'accès :cdev, cserv)
prop (COMPTE --> CORESPONDANT)
poss (ADRESSE SWIFT --> CORRESPONDANT)

Application 8

8.1 Description

```
FOR EACH p = PAYS

    DO sortir nom français (p) ;
       sortir nom néerlandais (p) ;
       sortir renseignements (p) ;
       FOR EACH v = VILLE FROM p VIA cont

         DO sortir nom (v) ;
           FOR EACH C = CORRESPONDANT FROM v VIA loc

              DO FOR b/p = BANQUE/PAYS FROM C VIA est reprt

                   DO sortir raison sociale (b/p);
                   OD
              FOR EACH cpt = COMPTE FROM C VIA prop

                   DO sortir numéro (cpt);
                     sortir cdev (cpt);
                     sortir renseignements (cpt)
                   OD;
              OD;
         OD;
    OD;
```

8.2 Quantification (2 x par an)

Nombre de pays : 200
Nombre de villes : 1 000 ==> 5 villes en moyenne par pays.
Nombre de correspondants : 6 000 ==> 6 correspondants en moyenne par ville
Nombre de comptes : 5 200 = 0.9 par correspondant en moyenne.

8.3 Détail des opérations

```
{200 x { Accès PAYS
            5 x { Accès ville (cont)
                  6 x { Accès correspondant (loc)
                        1 x { Accès BANQUE/PAYS (est reprt)}
                        0.9 x { Accès COMPTE (prop) }}}}}
```

8.4 Bilan des opérations (2 x par an)

ACCES PAYS	200
ACCES VILLE (cont)	1 000
ACCES CORRESPONDANT (loc)	6 000
ACCES BANQUE/PAYS(est reprt)	6 000
ACCES COMPTE (prop)	5 200

8.5 Structures d'accès mises en oeuvre

Chemin d'accès :

cont (PAYS --> VILLE)
loc (VILLE --> CORRESPONDANT)
est reprt (CORRESPONDANT --> BANQUE/PAYS)
prop (CORRESPONDANT -->COMPTE)

Application 9

9.1 Description

```
FOR EACH b = BANQUE

    DO sortir raison sociale (b) ;
    FOR EACH b/p = BANQUE/PAYS FROM b VIA repr

        DO FOR p = PAYS FROM b/p VIA comp

            DO sortir nom français (p) ;
               sortir non néerlandais (p) ;
            OD
              sortir raison sociale (b/p) ;
              sortir renseignements (b/p) ;
              FOR EACH C = CORRESPONDANT FROM b/p VIA est
            reprt

                DO FOR v = VILLE FROM C VIA loc

                    DO sortir nom (v) ;
                    OD
                    FOR EACH cpt = COMPTE  FROM  C  VIA
                    prop

                        DO sortir numéro (cpt) ;
                          sortir renseignements  (cpt)
                        ;
                        OD

                OD

        OD
    OD
```

9.2 Quantification (2 x par an)

Nombre de banques : 2 000
Nombre de banques/pays = 3 000 => nombre de B/P par banque = 3 000 : 2 000 = 1.5
Nombre moyen de correspondants par banque/pays = 6 000 : 3 000 = 2
Nombre moyen de comptes par correspondant : 5 200 : 6 000 = 0.9

9.3 Détail des opérations

```
{2 000 x ( Accès Banque
           1.5 x {Accès BANQUE/PAYS (repr)
                  1 x {Accès PAYS (comp) }
                  2 x {Accès  CORRESPONDANT (est reprt)
                  1 x {Accès VILLE (loc)}
                0.9 x {Accès COMPTE (prop) }}}}}
```

9.4 Bilan des opérations (2 x par an)

ACCES BANQUE	2 000
ACCES BANQUE/PAYS (repr)	3 000
ACCES PAYS (comp)	3 000
ACCES CORRESPONDANT (est reprt)	6 000
ACCES VILLE (loc)	6 000
ACCES COMPTE (prop)	5 200

9.5 Structures d'accès mises en oeuvre

Chemin d'accès :

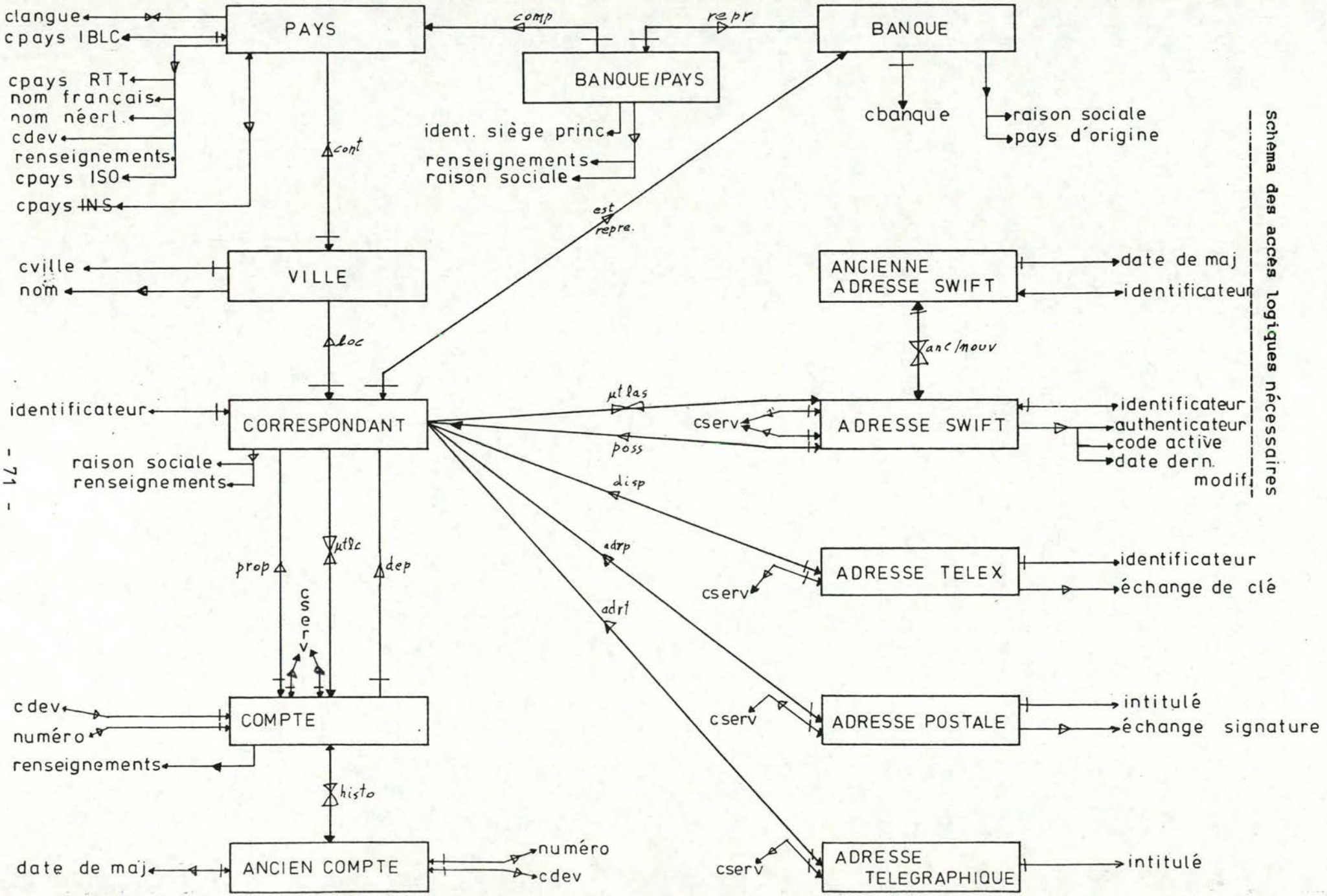
```
repr (BANQUE --> BANQUE/PAYS)
comp (BANQUE/PAYS --> PAYS)
est reprt (BANQUE/PAYS --> CORRESPONDANT)
loc (CORRESPONDANT --> VILLE)
prop (CORRESPONDANT --> COMPTE)
```

Bilan journalier

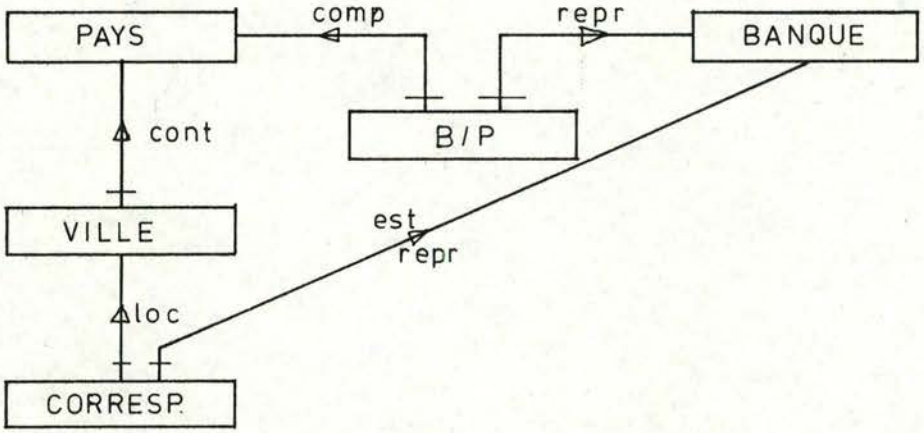
Accès compte (numéro,cdev)	1 567
Accès correspondant (prop)	2 631
Accès adresse swift (poss)	1 064
Accès adresse swift (utlas)	304
Accès adresse télex (Disp)	120
Accès correspondant (identificateur)	94
Accès adresse télégraphique (adrt)	28
Accès adresse postale (adrp)	42
Accès banque / pays (est reprt)	47
Accès ville (loc)	47
Accès pays (CINS)	11
Accès pays (cpays IBLC)	10
Accès adresse swift (identificateur)	1 520
Accès correspondant (poss)	1 520
Accès compte (prop)	456
Accès compte (utlc)	1 064
Accès correspondant (dep)	1 520

+2 x par an

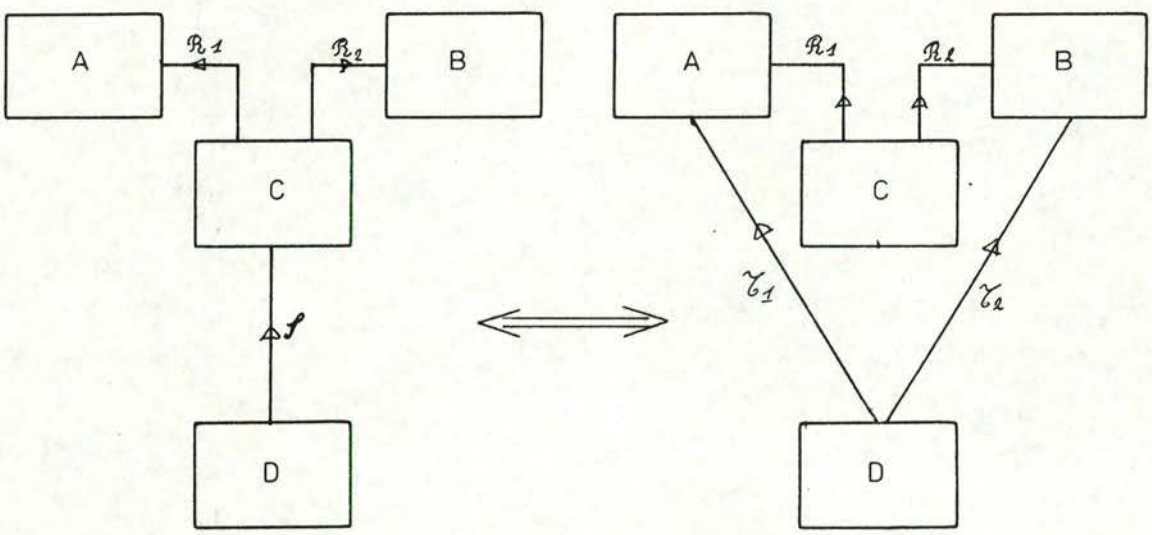
Accès pays (sequent)	200
Accès ville (cont)	1 000
Accès correspondant (loc)	6 000
Accès banque / pays (est reprt)	6 000
Accès compte (prop)	5 200 + 5 200
Accès ville (loc)	6 000
Accès correspondant (est reprt)	6 000
Accès pays (comp)	3 000
Accès banque (sequentiel)	2 000
Accès banque / pays (repr)	3 000


clangue
cpays IBLC
PAYS
comp
repr
BANQUE
BANQUE/PAYS
cpays RTT
nom français
nom néerl.
cdev
renseignements
cpays ISO
cpays INS
cont
ident. siège princ
renseignements
raison sociale
cbanque
raison sociale
pays d'origine
est repre.
cville
nom
VILLE
loc
ANCIENNE ADRESSE SWIFT
date de maj
identificateur
anc/mouv
identificateur
CORRESPONDANT
raison sociale
renseignements
utlas
cserv
poss
ADRESSE SWIFT
identificateur
authenticateur
code active
date dern. modif.
disp
adrp
adrt
ADRESSE TELEX
identificateur
échange de clé
cserv
prop
utlc
dep
cserv
ADRESSE POSTALE
intitulé
échange signature
cserv
cdev
numéro
renseignements
COMPTE
histo
ANCIEN COMPTE
date de maj
numéro
cdev
ADRESSE TELEGRAPHIQUE
intitulé
cserv
Schéma des accès logiques nécessaires

Dans l'application 4, nous remarquons que nous avons besoin d'un chemin d'accès entre CORRESPONDANT et BANQUE/PAYS. En effet, au point de vue conceptuel, l'information existe. Nous pouvons retrouver la banque / pays d'un correspondant en déterminant son pays d'origine (p) par (CONT O LOC) et sa banque d'origine (b) par (EST REPRT). Ayant identifié ce pays et cette banque, le couple (b,p) identifie une banque/pays. Nous pourrions cependant décider d'offrir un type de chemin direct en transformant le schéma des accès . Rappelons d'abord le schéma initial réduit aux types d'articles qui nous intéressent.



Nous connaissons une transformation qui est présentée par ailleurs et que nous rappelons ici.



cfr. contraintes en annexe 5.

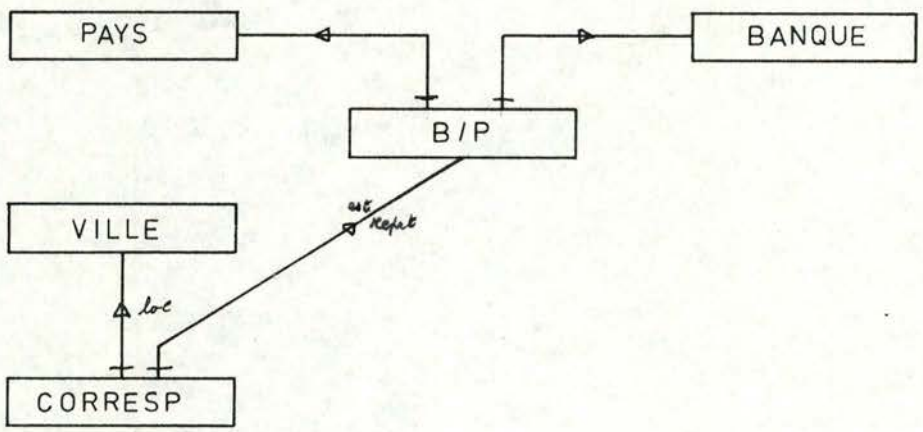
En considérant la composition de CONT avec LOC comme étant le type de chemin SITUE

Examinons si les contraintes sont vérifiées :
il faut que 1) EST REPR [c] = SITUE [c]
2) SITUE o COMP c EST REPR o REPR

- La première contrainte est vérifiée par le fait que les relations EST REPR et SITUE sont toutes les deux fortes pour correspondant et qu'elles identifient ensemble un correspondant (contrainte 3).

- La deuxième signifie que l'ensemble des pays (p) des correspondants de l'ensemble des banques (b) doit être inclus dans l'ensemble des pays situant les banque/pays de l'ensemble des banques. Ce qui est vérifié par la contrainte 4.

Nous obtenons donc le schéma suivant :



Nous remarquons cependant que nous avons perdu de l'information en supprimant le type de chemin CONT. En effet, nous ne pouvons plus déterminer les villes appartenant à un pays. Comme nous avons besoin de cette information , nous conserverons ce type de chemin.

QUATRIEME PARTIE

Quatrième partie : Expression des structures de données et des primitives d'accès DMS dans le modèle d'accès.

A. LES STRUCTURES DE DONNEES.

Base de données :

A ce concept, appelé DATA BASE, est associé un nom qui identifie une base de données parmi l'ensemble des bases de données disponibles.

Une DATA BASE est décrite par son nom
COMPILE‹data base-name›WITH DASDL LIBRARY

ARTICLE (TYPE D')

Les articles (RECORD) sont répartis en type (DATA SET) identifiés par un nom au sein d'une même base de données.

-‹data set name› _____DATA SET _____!

ARTICLE SYSTEME

Il existe un concept DMS II (appelé global data) qui correspond à cet article ; il lui correspond un ensemble d'items qui contiennent des informations communes à toute la base. Elles peuvent être accédées de la même manière que les items des types d'articles. Son nom est en fait le nom de la base de données.

____ ‹global item ›_____!
!__ ‹data item ›____!
!__‹group item ›____!
!__‹population item›__!

ITEM (VALEUR D'ITEM)

A un type d'article peut être associés 0,1 ou plusieurs items.
Un item peut être élémentaire ou décomposable (GROUP, FIELD).
Il peut être simple ou répétitif (OCCURS)
Il peut être de répétitivité fixe :
OCCURS < integer > TIMES
ou il peut être de répétitivité variable avec limite maximale :
- OCCURS < integr > TIMES
  DEPENDING ON < item - name>
Il peut prendre une valeur indéterminée (NULL).

Un item est déclaré comme composant d'un type d'article ou d'un item décomposable. La description se présente dans l'ordre dynastique de la structure de de décomposition. (cfr Déclaration COBOL /PL1)
Les numéros de niveaux n'apparaissent pas mais peuvent être déduits de la structure de parenthèses.

Exemple :

```
PERSONNE DATA SET     01 PERSONNE
(NOM                  02 NOM
 PRENOM               02 PRENOM
 ADRESSE GROUR        02 ADRESSE
 (RUE                 03 RUE
  NUMERO   <==>       03 NUMERO
  LOCALITE GROUR      03 LOCALITE
   (NUM-POSTAL        04 NUM-POSTAL
    COMMUNE           04 COMMUNE
    )
  )
)
```

CHEMIN D'ACCES (TYPE DE)

DMS II offre plusieurs mécanismes distincts pour traduire un type de chemin. Nous allons passer en revue ces mécanismes en précisant la cardinalité que chacun d'eux supporte.

I Imbrication (1 - N)

L'imbrication correspond à un type de chemin 1 - N.
Le type d'article cible est déclaré à l'intérieur de la définition du type d'article origine. Le type de chemin est automatique et obligatoire.
La structure définie par l'ensemble de ces types de chemin doit être arborescente.

```
<data set name 1> ____ DATA SET ;___!
( ---
<data set name 2> ____ DATA SET ;___!
    (
    ) ;
  ) ;
1 = type d'article origine du type de
    chemin
2 = type d'article cible du type de
    chemin
```

II Liens (N - 1 ; M - N)

Ce mécanisme correspond à deux types de chemin a) N -1 (liens)
b) M - N (liens avec clause OCCURS)

a) (N - 1)
Ce type de chemin est réalisé par un lien entre deux types d'articles.
Le lien est déclaré dans le type d'article origine du type de chemin et référence le type d'article cible du type de chemin.
Nous donnerons la règle d'existence des liens à la fin de cette rubrique sur les types de chemin

```
<data set name 1> _____ DATA SET ; ___!
   ( ---
<link name> __ REFERENCE TO _<data set
 name 2> ; ___!
 ---
 ) ;
1 = type d'article origine du type de
    chemin
2 = type d'article cible du type de
    chemin
```

b) (M-N)
Le type de chemin est réalisé par un lien entre deux types d'article.
Le lien est déclaré dans le type d'article origine du type de chemin et référence le type d'articles cible du type de chemin.

```
<data set name 1> ____ DATA SET ; ___!
  ( ---
<link name> - REFERENCE TO -<data set
  name 2) -->
  ->--- OCCURS <integer> TIMES;___!
   ) ;
1 = type d'article origine du type de
    chemin
2 = type d'article cible du type de
    chemin
```

La clause (occurs) est obligatoire.

III Subset manuel (M - N)

Le type d'article origine du type de chemin est celui dans lequel est déclaré le subset et le type d'article cible est celui référencé par ce subset

```
<data set name 1> ____ DATA SET;___!
  ( ---
<subset name> - SUBSET OF - <data set
  name 2> ___!
   ) ;
```

Règle d'existence des liens et des subsets manuels

Le type d'article père du type d'article cible du lien ou du subset manuel doit être un ancêtre (père, grand-père,...) du type d'article origine. On détermine l'ascendance d'un type d'article en utilisant l'arborescence définie par les imbrications de types d'article. L'article système est considéré comme le sommet de cette arborescence.

CLE D'ACCES

DMS offre 3 mécanismes d'accès qui, suivant le niveau de leur définition porte sur un référentiel différent. Par niveau, on entend le fait que le type d'article référencé par le mécanisme soit disjoint (fils de l'article SYSTEME) ou soit imbriqué (descendant d'un type d'article disjoint).

1) Access

Mécanisme qui organise les articles d'un type sur base des valeurs de la ou des clé énoncées. Si l'ACCESS est déclaré sur un type d'article disjoint, le référentiel sera tous les articles de ce type. Si le type d'article est imbriqué, le référentiel sera l'ensemble des articles cible du chemin défini par l'imbrication. Un seul ACCESS peut être déclaré sur un même type d'article.

```
<access name> ACCESS TO
  <data set name>
 KEY IS <item name,.....>
```

2) set

Mécanisme externe au type d'article qui définit une clé d'accès aux articles du type qu'il référence. Le référentiel est obtenu de la même manière que pour l'ACCESS. Plusieurs SETS peuvent être déclaré sur un même type d'article.

```
<set name> SET OF <data set name>
```

3) Subset

Mécanisme semblable au SET à la différence que le référentiel est dans ce cas soit un sous-ensemble des articles du type référencé par le SUBSET soit un sous-ensemble des articles cibles du chemin défini par l'imbrication

```
<subset name> SUBSET OF
  <data set name>
  WHERE <boolean expression>
  KEY IS <item name,.....>
```

FICHIERS

A un set, subset, data set est assigné un fichier. Néanmoins, l'utilisateur peut modifier cette assignation en utilisant la clause PARTITION qui lui permet d'assigner plusieurs fichiers à un set, subset, data set.

ORDRE DES ARTICLES CIBLES D'UNE CLE D'ACCES

Les articles associés à une clé d'accès non identifiante (DUPLICATES) sont accessibles dans l'ordre chronologique (FIFO) ou antéchronologique (LIFO). Ils sont également accessibles par valeurs croissantes ou décroissantes de clé.

```
- DUPLICATES _ LAST _____!
             !_FIRST__!
__ ASCENDING _____!
 !_DESCENDING ___!
```

CHEMINS D"ACCES IMPLICITES

Un accès séquentiel à tous les articles d'un même type est offert alors qu'aucun chemin d'accès n'a été déclaré explicitement.

CONTRAINTES D"EXISTENCE ASSOCIEE A UN TYPE DE CHEMIN.

Type de chemin fort ou faible

L'imbrication offre un type de chemin fort pour le type d'article cible du chemin. Pour les autres mécanismes (Liens - Subsets), les types de chemin sont faibles pour l'origine comme pour la cible.

Mode d'appartenance

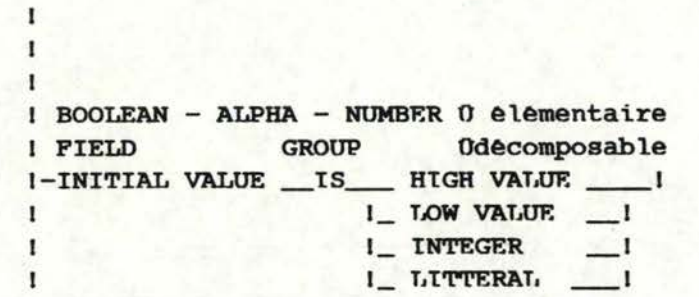
L'imbrication offre un mode d'appartenance automatique, fort et fixe pour le type d'article cible. Les autres mécanismes (Liens - subsets) offrent un mode d'appartenance manuel, facultatif et faible pour le type d'article cible.

CONTRAINTES SUR LES VALEURS D'ITEM

On peut spécifier l'image d'un item (élémentaire - décomposable)

On peut leur attribuer une valeur initiale

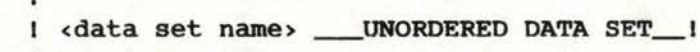
On peut spécifier la plage de valeurs que peut prendre un item.
La clause VERIFY déclarée au niveau du type d'article permet de vérifier que l'item mentionné par cette clause vérifie l'expression énoncée.



ORDRE DES ARTICLES CIBLES D'UN CHEMIN D'ACCES

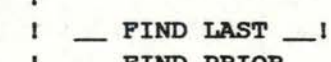
Ordre chronologique (FIFO)

En utilisant l'organisation UNORDERED pour le type d'article cible du type de chemin, les articles sont rangés dans l'ordre de leur introduction.



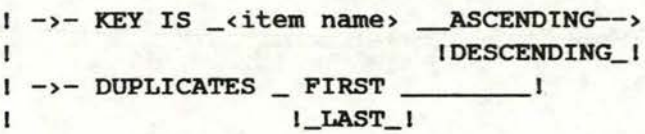
Ordre antéchronologique (LIFO)

Il n'existe pas en DMS mais il peut être trouvé en utilisant le même type d'organisation et en lisant les articles en sens inverse du dernier vers le premier.



Ordre trié selon une valeur de clé

On peut déclarer une clé d'accès sur un type de chemin qui permet d'obtenir les articles cibles d'un article origine selon les valeurs de cette clé. On peut les obtenir dans l'ordre des valeurs croissantes (ASCENDING) ou décroissantes (DESCENDING) des valeurs de clé. Si les doubles sont autorisés (DUPLICATES), ils peuvent être obtenus dans l'ordre FIFO (LAST) ou LIFO (FIRST). Dans le cas d'un type de chemin réalisé par l'imbrication de deux types d'article, on définira la clé soit par un ACCESS et le type d'article cible sera déclaré ORDERED, soit par un set défini sur le type d'article cible. Dans le cas d'un type de chemin réalisé par un subset , la clé sera déclarée dans la définition de celui-ci.



CONFIDENTIALITE

Un fichier gardien est créé par base de données et contient par utilisateur le nom des programmes que celui-ci peut exécuter , le genre d'accès (ECRITURE, LECTURE, ECR/LECT) qu'il peut exécuter sur la base.

Notion de sous-schéma

Celle-ci est réalisé en DMS par deux notions : la base logique et les "REMAPS DATA SET"

La base logique permet de :

- réduire le nombre de types d'articles accessibles
- restreindre le type d'accès
- empêcher l'exécution de certaines commandes

Les REMAPS DATA SET permettent de

- changer le nom des types d'article (REMAP)
- changer l'ordre et la représentation des items
- réduire le nombre d'items par type d'article
- réduire la plage de valeurs de certains items (SELECT, VERIFY)
- placer des verrous de lecture unique sur certains items (READ ONLY)
- utiliser des items dont la valeur restera cachée (HIDDEN) (intéressant pour les données d'accès)

<logical data base name> LOGICAL DATA BASE,DATA SET (<data set name>,...)

Un fichier gardien est associé à chaque base logique.

<remap name> REMAP <data set name>

__ SELECT ____ <boolean expression >____!
!_ VERIFY__!

_ <item name >_____READ ONLY___!

_ <item name >_____HIDDEN_____!

Influence d'un sous-schéma sur les primitives

Un programme d'application travaillant sur une base logique, les seules opérations permises sur les données pour un programme sont celles qui :

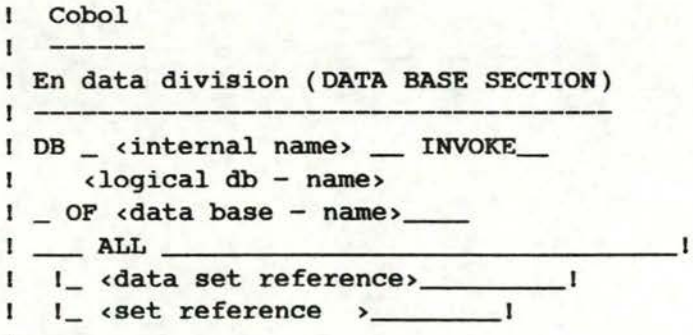
- concernent les objets décrits dans la base logique (sous-schéma)
- respectent les types d'accès permis et qui sont décrits dans le fichier gardien et au niveau de chaque item.

Parmi celles-ci , celles pour lesquelles toutes les contraintes d'intégrité déclarées dans la base logique peuvent être respectées.

Communications entre programme utilisateur et SGBD

Définition du contexte

Tout programme destiné à travailler sur une base de données doit en invoquer une base logique. Il peut en invoquer toutes les structures soit un certain nombre de celles-ci. Un même programme peut invoquer plusieurs bases logiques



Références aux objets de la base

Le mode fondamental de référence aux articles et aux chemins d'accès à la disposition du programme est le chemin courant (CURRENT PATH)
Les chemins courants sont des pointeurs vers des articles de la base .
Ils sont mis à jour par le SGBD lors de l'exécution des primitives d'accès et de certaines primitives de mise à jour.
Les différents chemins courants sont :

- Pour chaque type d'article,
  il existe un chemin courant qui est la référence du dernier article de ce type auquel le processus a accédé ou qu'il a créé.
- Pour chaque set,
  il existe un chemin courant qui est la référence au dernier article accédé par ce set.
- Pour chaque subset automatique ou manuel

il existe un chemin courant qui est la référence au dernier article accédé par ce subset.

Un chemin courant est indéterminé si aucun article qu'il peut référencer n'a été accédé jusqu'à présent et invalide si l'article qu'il référençait a été supprimé. Il faut savoir que chaque utilisateur d'une même base a ses propres chemins courants.
Un même utilisateur peut disposer de plusieurs chemins courants par type d'article, par set et subset en invoquant plusieurs copies de ces types d'article ,de ces sets,de ces subsets.

La zone de communication

Tout programme travaillant sur un base de données doit avoir une zone de communication avec celle-ci qui permettra les échanges. Cette zone conviendra : — USER WORK AREA (UWA)

- pour chaque type d'article invoqué , un espace (RECORD AREA) destiné à recevoir les valeurs des items de ce type d'article ; l'organisation et la désignation des composants de cet espace obéissent aux règles COBOL relatives aux structures de RECORD. Cette zone constitue un "tampon logique" pour le type d'article. L'article se trouvant dans cette area sera appelé article courant du type d'article auquel il appartient.
- pour chaque set,subset invoqué, un espace destiné à recevoir les valeurs d'items du type d'article sur lequel le set ou le subset est défini, on appellera cet espace article courant du set ou du subset.
- un registre spécial (DMSTATUS) qui indique s'il y a lieu :
  - la catégorie d'erreur — (NOT FOUND, DUPLICATES, DEADLOCK,...)
  - son type
  - la structure concernée (type d'article,set,subset,...)

B. LES PRIMITIVES

Préambule

Il existe deux modes de travail sur la base de données. Soit en se servant des primitives qui sont insérées dans des programmes COBOL, PL/1, ALGOL, soit en se servant du langage interactif DMS II inquiry. Ce dernier mode ne permet que la consultation et la mise à jour d'items. Nous ne présenterons pas ici les primitives de ce langage.

Nous classerons les primitives d'accès en :

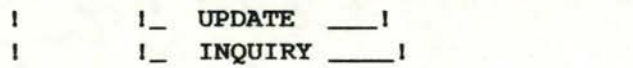
- Accès à la base de données ; — OPEN, CLOSE
- Accès à un article ; — FIND
- Accès à des valeurs d'item ; — GET, MOVE

OUVERTURE DE LA BASE

argument :

nom de la base de données < db - name >

mode d'accès
- consultation (INQUIRY)
- consultation + modification (UPDATE)
- création de la base (INITIALIZE)



fonction :

Cette primitive ouvre la base de données < db - name> dans le mode d'accès en s'assurant que le programme l'exécutant a les droits d'accès requis

FERMETURE DE LA BASE

argument :

nom de la base à fermer < db name >

fonction :

Cette primitive ,après avoir libérer les éventuels articles encore verrouillés , ferme la base < d-b name >.
Cette primitive est optionnelle car le système fermera les bases de données ouvertes à la terminaison du programme les ayant ouvertes.

__CLOSE__ <d-b- name ____

ACCES AUX ARTICLES CIBLES D"UN CHEMIN

ACCES AU PREMIER (DERNIER)

arguments :

c : soit (a) manuel subset name
courant du type d'article origine du type de chemin
soit (b) data-set-name (imbrication)
courant du type d'article origine du type de chemin

(a) FIND ___ FIRST _____<subset name>___
!_ LAST ____!

fonction :

Cette primitive fournit le premier (dernier) article cible du chemin c qui peut être défini des deux manières présentées ci-dessus suivant que nous avons affaire à un type de chemin déclaré par un subset manuel (a) ou par un type d'article imbriqué (b)

(b) FIND ___ FIRST _____<data set name>___
!_ LAST ____!

ACCES AU SUIVANT (PRECEDENT)
---

arguments :
---

c : soit (a) - manual subset name
- courant du type d'article origine du type de chemin
p. courant du subset manuel

(b) - data set name (imbrication)
- courant du type d'article origine du type de chemin
p. courant du type d'article cible du type de chemin

fonction :

Cette primitive fournit, s'il existe, l'article cible du chemin c qui est le suivant (précédent) de p. Dans le cas (a) le type de chemin est un subset manuel Dans le cas (b), il s'agit d'un type d'article imbriqué.

(a) FIND ___NEXT ____< subset name >____!
!_PRIOR__!

(b) FIND __NEXT___<data set name>__!
!_PRIOR_!

ACCES DIRECT DANS UN CHEMIN
---

arguments.
---

c : link name (avec clause OCCURS dans le schéma)
- courant du type d'article origine du type de chemin
i : entier
At : data set name0

fonction

Cette primitive a pour but de fournir le ième article cible du chemin c si t contient au moins i articles cibles. Il faut savoir que l'ordre des articles cibles d'un chemin défini par un link ne reste pas fixé une fois pour toutes. La possibilité est cependant donnée au programmeur de gérer lui-même cet ordre.

FIND __< link name > __ (i)___!

ACCES PAR CLE

ACCES AU PREMIER (DERNIER) ARTICLE ASSOVIE A UNE VALEUR DE CLE

arguments :

c : a) automatic subset name b) set name
k : noms des items (it1, it2,...) déclarés dans la clause KEY IS de c
v : contenu des it1, it2,... en UWA
[t : data set name]

(a) FIND__FIRST___<subset name>_AT_ k=v__!
!_LAST__!

(b) FIND ___FIRST ___<set name>_AT_k=v__!
!_LAST__!

fonction :

Si k est une clé d'accès définie sur t, cette primitive fournit ,s'il existe, le premier (dernier) article du type t qui satisfait à la condition exprimée sur la clé (k = v)

ACCES A L'ARTICLE SUIVANT (PRECEDENT) ASSOCIE A UNE VALEUR DE CLE

arguments :

c : a) automatic subset name b) set name
k : noms des items (it1, it2...) déclarés dans la clause KEY IS de c
v : contenu de it1,it2... en UWA
p : courant du -a)subset automatique b)set
[t : type d'article]

(a) FIND __NEXT___<subset name>_AT_k=v___!
!_PRIOR_!

(b) FIND __NEXT__<set name>__AT_k=v___!
!_PRIOR_!

Fonction

Si k est une clé d'accès définie sur t, cette primitive fournit l'article

du type t, s'il existe, qui satisfait à la condition exprimée sur la clé (k = v) et qui est le suivant (précédent) de p.

ACCES PAR CLE DANS UN CHEMIN (MANUAL SUBSET - TYPE D'ART.IMBRIQUE)
(a) (b)

Accès au premier (dernier)

arguments :

C : (a) - manual subset name
- courant du type d'article origine du type de chemin

(b) - data set name
- courant du type d'article origine de ce type de chemin

k : noms des items (it1,it2 ...) déclarés dans la clause KEY IS
(a) du subset manuel
(b) de l'ACCESS défini sur le type d'article cible du type de chemin

v : Contenu de ces items (it1,it2 ...) en UWA[t : data set name]

fonction

Cette primitive fournit, s'il existe, le premier (dernier) article cible du type t du chemin c satisfaisant à la condition exprimée sur la clé (k = v)

(a) FIND __FIRST____<subset name>AT_k=v_!
!_LAST__!

(b) FIND __FIRST___<data set name>_AT k=v_!
!_LAST__!

Accès au suivant (précédent)

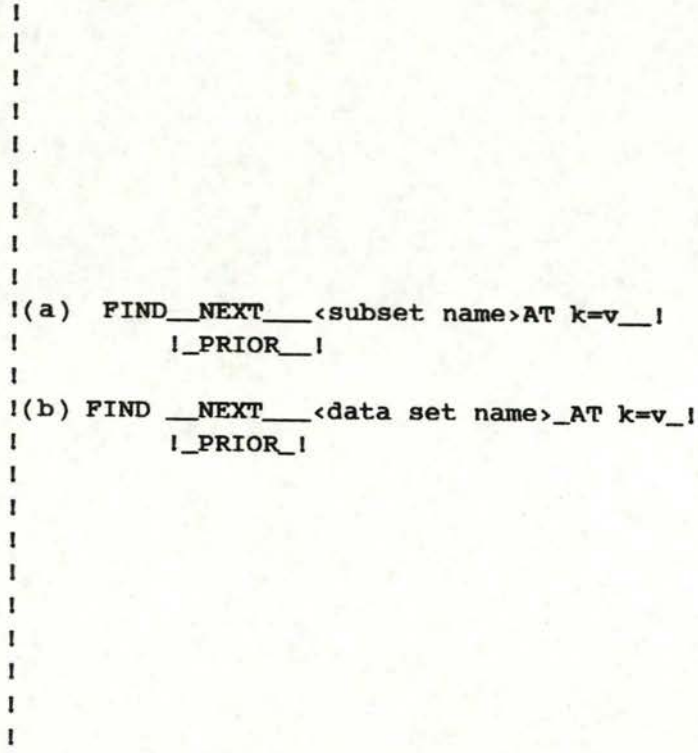
arguments :

C : (a) manal subset name
courant du type d'article origine du type de chemin

(b) data set name
courant du type d'article origine du type de chemin
k : cfr accès au premier
v : cfr accès au premier
p : (a) courant du subset
(b) courant du type d'article cible du type de chemin
[t : data set name]

fonction :

Cette primitive fournit, s'il existe, l'article cible du type t du chemin c satisfaisant à la condition suivante exprimée sur la clé (k = v) et suivant (précédent) de p.



ACCES AUX VALEURS D'ITEMS

Ces valeurs sont disponibles aux programmes COBOL / PL1 dès que l'article qui les contient a été accédé (FIND).
L'ordre FIND a en effet pour but de transférer les valeurs d'items dans la RECORD AREA de l'user work area après l'avoir trouvé. Pour les programmes ALGOL , le GET est utilisé pour transférer l'information de UWA dans des variables ou des tableaux ALGOL.

arguments :

- t : data set name
- variables ALGOL
- items de t

GET_<data set name>__<variable : =
                      <data item>

Traitement des conditions d'exception

Chaque primitive comprend une clause d'exception qui permet d'exécuter une procédure nommée dans le cas où la primitive qui contient cette clause n'a pas réussi.

ON EXCEPTION _<statement -1>
                <else statement -2>

Les déclaratives COBOL peuvent inclure une procédure qui sera exécutée en cas d'erreur sur une des primitives qui ne comporterait pas de clause d'exception

USE ON DMERROR
 procédure,

LES PRIMITIVES DE MODIFICATION

INITIALISATION D'UN ARTICLE

argument :

- t : data set name

fonction :

Cette primitive sert à initialiser l'UWA.
Les items d'une RECORD area sont initialisés avec les valeurs "initiales" déclarées dans le schéma, à défaut , la valeur "NULL".

CREATE __<data set name>____

MISE A DISPOSITION D'UN ARTICLE DES VALEURS D'ITEM

arguments :

- t : data set name
- it1,it2 ... items de t
- expression exprimée en termes de variables ALGOL.

fonction :

Cette primitive permet de transférer l'information se trouvant dans des variables ALGOL vers la record area du type d'article t

PUT_<data set name>__<data item> : =
<expression>

CREATION D'UN ARTICLE

Arguments :

- t : data set name
- it1,V1 : tous les items déclarés pour t dans la base logique
- it2,V2 : le contenu dans l'UWa de l'espace réservé à ces items
C1 ensemble des sets et subsets automatiques dans lesquels l'article doit être
: inséré.

Fonction :

Si les contraintes d'intégrité définies dans le schéma peuvent être vérifiées à l'aide des arguments, le SGBD enregistre dans la base de données , un article auquel sont associées les valeurs Vi des items it i.
Il l'ajoute aux sets et aux subsets dont il doit faire partie. Si tout a réussi ,alors le chemin courant du type d'article référence ce nouvel article.

STORE __<data set name>_____!

SUPPRESSION D'UN ARTICLE

arguments :

(a) t : data set name
(b) expression de sélection de l'article (cfr accès aux articles (FIND))

DELETE ___<data set name>____!

!_<selection expresssion>__!

fonction :

(a) Cette primitive retire de la base l'article courant du type d'article t
(b) Cette primitive retire de la base l'article qui satisfait à l'expression de sélection.
Cet article doit être verrouillé (LOCK). Ce verrouillage est réalisé

implicitement dans le cas (a). Dans l'autre cas (b),il faut réaliser un LOCK.
Si cet article est origine d'un chemin non vide , il ne peut être supprimé.
Il sera retiré de tous les sets ou subsets automatiques auxquels il appartient
par la primitive.

MODIFICATION DES VALEURS D'ITEMS

Cette primitive n'existe pas telle qu'elle en DMS. Elle est en fait la séquence de plusieurs autres.(LOCK _ PUT _ STORE _ FREE)

1) Verrouillage de l'article (LOCK)

arguments :

(a) t :data set name
(b) expression de sélection
(cfr accès aux articles FIND)

LOCK _____<data set name>_____
_<selection expression>__

fonction

Cette primitive verrouille,s'il n'est pas déjà verrouillé, éventuellement par un autre utilisateur, l'article courant (a) du type d'article t ou l'article qui répond à l'expression de sélection (b)
On modifiera les valeurs d'items comme défini auparavant (PUT pour l'algol, MOVE pour COBOL, = pour PL1) ensuite on replacera l'article dans la base données (STORE).
Ensuite, il faut déverrouiller l'article (FREE)

Le STORE reconnaîtra qu'il ne s'agit pas d'une création car cet article est verrouillé. (LOCK)

2) Déverrouillage de l'article

arguments :

t : data set name

__FREE __<data set name>_____

fonction :

Cette primitive déverrouille l'article courant du type d'article t.

INSERTION D'UN ARTICLE DANS UN CHEMIN

arguments :

- a : courant du type d'article cible (t)
- t : data set name

C : - (a) subset name
    - (b) link name
    - (c) t : data set name (type d'article imbriqué)

fonction :

(a) SUBSET MANUEL

INSERT_<data set name>_INTO<subset name>

Cette primitive insère l'article a dans le chemin c et le chemin courant du subset référencera l'article inséré. L'insertion de l'article a ne peut vider les contraintes exprimées sur c.

(b) LINK

Cette primitive insère l'article a dans le chemin c. L'article origine du chemin, celui qui contient la définition de link name , doit avoir été verrouillé. L'insertion de l'article a ne peut vider les contraintes exprimées sur c.

ASSIGN <data set name>_TO_<link name>__

(c) IMBRICATION

C'est en fait la première création qui réalise l'insertion dans c. En effet, le mode d'appartenance défini par l'imbrication est forte, fixe et automatique.

STORE __<data set name>_____

RETRAIT D"UN ARTICLE DANS UN CHEMIN

arguments :

a : (a) courant du subset
    (b),(c) courant du type d'article cible du type de chemin
t : data set name
C : - (a) subset name
    - (b) link name
    - (c) t (type d'article imbriqué)

fonction :

(a) Cette primitive supprime l'article a du chemin c .Si l'opération réussit le chemin courant du subset devient invalide

REMOVE__CURRENT__FROM <subset name>__

(b) Cette primitive supprime l'article a du chemin c .Si le lien est "COUNTED" le nombre d'article cible (COUNT) restant de c est mis à jour .

ASSIGN__NULL__TO__<link name>___

(c) C'est en fait la primitive de suppression d'un article qui réalise cette suppression. En effet le mode d'appartenance des articles cibles d'un chemin est fixe, automatique et forte dans le cas de l'imbrication.

DELETE _<data set name>_____

MODIFICATION DES CHEMINS COURANTS

---

argument :

---

t : data set name
s : set name

fonction :

Cette primitive modifie le chemin courant d'un type d'article ou d'un set en le forçant à référencer le début (fin) du type d'article ou du set S. Le chemin courant ne référencera pas le premier (dernier) article mais un FIND NEXT (PRIOR) consécutif à cette primitive aura cet effet.

SET <data set name>__TO__BEGINING____
!_<set name>_____! !_ENDING___!

Argument :

---

t : data set name
S : set name
a : article courant de t

Fonction :

Cette primitive modifie le chemin courant de S qui référencera l'article courant de t (a).

SET <set name> __ TO _ <data set name>_!

CINQUIEME PARTIE

Cinquième partie : Propositions d'implémentation.

A Introduction

Dans cette partie, nous allons énoncer un certain nombre de critères qui vont nous servir à discriminer les différentes propositions que nous ferons pour implémenter les types d'articles et les types de chemin .

Dans un deuxième temps, il nous faudra examiner le temps d'accès requis pour chaque primitive ainsi que le volume global de la base de données. De cette analyse , nous pourrons tirer des conclusions sur les performances réelles de la proposition faite. Si celles-ci ne sont pas satisfaisantes, cela nous conduira à transformer de nouveau le schéma des accès et à faire de nouvelles propositions qui seront à leur tour évaluées jusqu'à l'obtention d'une solution satisfaisante
C'est à ce niveau également qu'apparaîtrait , si nécessaire, une analyse plus fine des paramètres physiques (taille des blocs, pages; nombre de blocs..) qui entraînerait des politiques d'accès différentes.
Ensuite , il faudra définir les types de mécanismes de reprises en cas d' incident. Le degré de concurrence désiré vis-à-vis de l'accès aux données devra être confronté aux possiblités de SGBD ce qui pourra également modifier certains paramètres physiques préalablement fixés.

Nous n'avons pas la prétention d'avoir relevé toutes les implémentations possibles ni de les avoir analysées en fonction de tous les critères possibles, mais plutôt d'avoir proposé celles qui nous semblaient les plus avantageuses et d'avoir choisi entre elles.
Cette étape nous semble être la plus intuitive de celles que nous avons rencontrées. Il est certain que l'expérience joue ici, un grand rôle et qu'un concepteur initié passera rapidement au réglage fin des paramètres physiques et aux mécanismes de reprise.

Nous n'avons pas ici réaliser , faute de temps , l'analyse précise du temps d'accès de chaque primitive et l'évaluation globale de la proposition ainsi que l'étape suivante, soit parce que nous ne disposions pas de toutes les informations soit par manque de temps. En effet, ce genre d'analyse est longue et imprécise pour qui n'a pas une connaissance approfondie du système de gestion de base de données employé et plus précisément de ces paramètres physiques.

B Enoncé des critères

Le premier critère que nous avons retenu est la possibilité d'extensions ou de modifications futures, étant donné que le système est encore dans sa phase de développement. Nous en avons d'ailleurs déjà tenu compte dans l'analyse qui a précédé en introduisant de nombreux items non utilisés dans les applications.

Un deuxième critère est la sécurité de la base de données. Nous ne nous attarderons pas sur lui, étant donné la particularité de la procédure de mise à jour (tous les deux mois). En effet, il suffira après chaque exécution de la procédure, de prendre une copie de la base de données, ce qui nous garantira contre la majorité des incidents pouvant survenir durant l'exploitation de la base.

Un troisième critère dont nous tiendrons compte sera les facilités offertes au programmeur d'applications. Ici, nous mettrons en évidence les contraintes prises en charge par le système.

Un quatrième critère que nous retiendrons est la performance du système. Nous analyserons ici, deux éléments : les volumes et le temps d'accès. Nous n'oublierons cependant pas qu'une solution performante et complexe est loin d'être meilleure qu'une solution simple et satisfaisante. Notre cinquième et dernier critère pourrait d'ailleurs s'énoncer ainsi : essayer de ne pas s'éloigner outre mesure du schéma conceptuel.

C Propositions d'implémentation des types d'articles

1 Type d'article PAYS

a) Caractéristiques

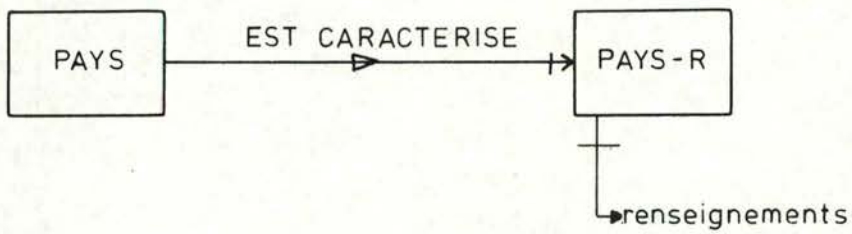
Article de taille moyenne (189 car.). On accède à l'item renseignements que peu souvent (édition de liste de correspondants (200 accès tous les six mois)). Les autres items soit sont beaucoup plus courts, soit sont accédés de façon plus régulière. Ce type d'article est relativement stable; le nom, les codes, la langue d'un pays changent peu ; les renseignements peuvent eux changer plus fréquemment. Une croissance quasi nulle est à envisager car tous les pays de la terre seront introduits dès le départ. Accès séquentiel pour la liste des correspondants par pays - ville.

b) Propositions

1) Un type d'article avec deux sets a1 et a2
a1 : clé identifiante : cpays IBLC
a2 : clé d'accès non identifiante : cpays INS

2) Deux types d'article : l'un contenant les renseignements, l'autre le reste des items et sur lequel seront définies les deux clés d'accès (sets)

Cette solution est obtenue ainsi :



Pour chaque article du type PAYS pour lequel l'item renseignements à une valeur significative, on crée un chemin d'accès EST CARACTERISE vers un article du type PAYS - R qui contient cette valeur significative. On créera un article de type PAYS - R pour chaque valeur significative différente de l'item renseignements du type d'article PAYS. Le type de chemin EST CARACTERISE de cardinalité (N-1) pourrait être implémenté par un lien (LINK). (cfr DMS - Modèle d'accès).

2 Type d'article VILLE

a) Caractéristiques :

Article court (23 car.). Accès régulier peu fréquent (47 fois par jour). Pour les listes des correspondants (7 000 accès tous les six mois). Type d'article relativement stable (modification pratiquement nulle, peu de croissance et de suppression)

b) Proposition

Un type d'article.
On pourrait également proposer un accès sur le code ville (clé identifiante)

3 Type d'article B/PAYS

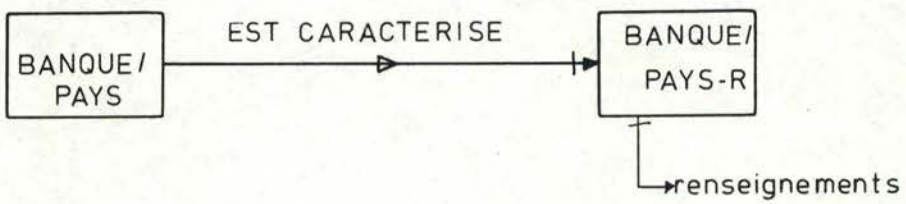
a) Caractéristiques :

Article de taille moyenne (204 car.) (47 fois par jour + 9 000 tous les six mois) Les renseignements (132 car.) ne sont accédés que lors de l'édition de la liste des correspondants par BANQUE (9 000 tous les six mois). Type d'article relativement stable (croissance assez faible, peu de modification et de suppression)

b) Proposition

1) Un type d'article

2) On peut proposer la même solution que pour le type PAYS.



Le type de chemin EST CARACTERISE de cardinalité (N-1) peut être implémenté par un lien (LINK). (cfr DMS - Modèle d'accès).

4 Type d'article BANQUE

a) Caractéristiques :

Article court (35 car.), accédé peu régulièrement (Liste des correspondants par BANQUE) (2 000 accès tous les six mois)
Type d'article stable.
Accès séquentiel aux banques sur base de leur nom.

b) Proposition

Un type d'article.

5 Type d'article CORRESPONDANT

a) Caractèristiques :

Article court (45 car.) d'accès frèquent (5 765 fois par jour + 1 200 tous les six mois).
Type d'article assez stable.
Clè d'accès identifiante (identificateur)

b) Proposition

Un type d'article avec un set qui dèfinit une clè d'accès identifiante : identificateur.

6 Type d'article ADRESSE SWIFT

a) Caractèristiques

Article court (37 car.)
Accès frèquent (2 888 par jour)
Accès sur base de l'identificateur.
Type d'article relativement stable.

b) Proposition

Un type d'article avec un set qui dèfinit une clè d'accès identifiante : identificateur.

7 Type d'article ADRESSE TELEX

a) Caractèristiques :

Article court (36 car.).
Accès peu frèquent (28 fois par jour).
Peu de mise à jour (suppression, crèation, modification)
Type d'article susceptible de contenir de nouveaux items dans un proche avenir.

b) Proposition

Un type d'article.

8 Type d'article ADRESSE POSTALE

a) Caractèristiques

Article court (47 car.)
Accès peu frèquent (42 fois par jour)
Peu de mise à jour.
Type d'article susceptible lui aussi, de contenir de nouveaux items dans un proche avenir.

b) Proposition

Un type d'article.

9 Type d'article COMPTE

a) Caractèristiques :

Article de taille moyenne (178 car.)
Accès frèquent (3 087 fois par jour) (+ 10 400 tous les six mois).
Peu de croissance et de modification.
Pas d'accès sèquentiel .
Accès par clè identifiante (num,cdev).
Les renseignements (132 car.) ne sont accèdès que lors de l'èdition de listes (10 400 fois tous les six mois)

b) Propositions

1) Un type d'article avec un set al
   al : clè identifiante : (num,cdev)

2) Deux types d'articles : l'un contenant les renseignements, l'autre le reste des items et sur lequel sera dèfini la clè d'accès.(cfr PAYS)

10 Type d'article ANCIENNE ADRESSE SWIFT

a) Caractèristiques

Article très court (19 car.)
Accès exceptionnel.
Application non encore décrite (statistiques).
Croissance très lente.
Aucun mode d'accès n'est requis car l'application n'est pas décrite , néanmoins on pourrait proposer une clé d'accès identifiante : identificateur.

b) Proposition

Un type d'article avec clé d'accès identifiante (identificateur).

11 Type d'article ANCIEN COMPTE

a) Caractèristiques

Article très court (23 car.)
Accès exceptionnel .
Application non encore décrite (statistiques)
Croissance très lente.
Aucun mode d'accès n'est requis mais on pourrait proposer (num,cdev) comme clé d'accès identifiante.

Proposition

Un type d'article avec clé d'accès (num, cdev).

Avant de proposer des implémentations pour les types de chemin, il nous faut ici revenir sur les différentes propositions et essayer d'établir un premier choix.

A trois reprises, nous avons rencontré le même problème : une différence significative entre l'accès à certains items d'un type d'article conjugué à une différence de taille non négligeable des articles peu accédés. Analysons les deux propositions en tenant compte des critères énoncés plus haut.

Porte ouverte aux modifications, extensions :

1) Adjonctions de nouveaux items :

On relève peu de différences. Avec la deuxième proposition, on pourra les adjoindre au premier ou au deuxième type d'article suivant le type d'accès qu'il requiert (fréquent - exceptionnel).

2) Adjonctions de nouveaux chemins :

Si l'item Renseignements devient une clé d'accès, la première proposition implique la création d'un set; la deuxième , la création d'un chemin d d'accès du type d'article contenant les renseignements vers le type d'article contenant les autres items. Ceci est un peu plus compliqué car les contraintes de cardinalité et d'existence [(1-N),fort pour l'origine] implique l'emploi d'un subset manuel ou d'un lien , les contraintes étant à charge des programmes de mise à jour. L'adjonction de type de chemin vers d'autres types d'article se comporte de la même manière

Performances (temps d'accès, place occupée)

I PAYS

- Si nous retenons une taille de bloc de 500 mots.

Dans le premier cas, on trouvera 500 mots : 32 mots (189 c) soit 15 articles PAYS par bloc. Les 200 pays tiendront dans 14 blocs.
Dans le deuxième cas, pays sans renseignements, (57 car = 10 mots + 1 mot de pointeur d'adresse (lien)), on trouvera 45 pays. Les pays se trouveront donc dans 5 blocs.
On trouvera dans un bloc , 22 valeurs de renseignements (22 mots) soit pour les 100 renseignements 5 blocs également.

a) Volume :
---

* Dans le premier cas , la déclaration de l'item renseignements dans le type d'article PAYS entraîne , pour cet item, un volume de 200 (nombre d'articles PAYS) x (189 car) (32 mots) soit 6 400 mots.

* Dans le deuxième cas, nous aurons 100 x (132 car. - 22 mots) = 2 200 mots pour les articles PAYS-R et 200 x (57 car = 10 mots) + 1 mot d'adresse soit 2 200 mots pour les PAYS, soit 4 400 mots au total.

b) Temps d'accès :

- Au point de vue journalier :
---

21 accès aléatoires aux articles pays sont requis. On n'accède pas à l'item renseignements. L'avantage est à la deuxième solution : en effet, après avoir accédé à un pays , la probablité de se retrouver dans le même bloc est de 1/5 soit 20 % , tandis que pour la première solution , elle sera de 1/14 soit 7 %.

- Au point de vue bi-annuel :

Pour lire l'ensemble des pays avec leurs renseignements dans le premier cas, il faudra lire 14 blocs et dans le deuxième cas, 10 blocs (accès séquentiel). L'avantage est donc, ici, à la deuxième solution , contrairement à ce que l'on aurait pu croire.

II BANQUE/PAYS :

En prenant les mêmes hypothèses que pour le type d'article PAYS, on trouve 14 articles BANQUE/PAYS par bloc pour la première solution.
Dans le deuxième cas, on trouvera 41 articles par bloc pour l'article BANQUE/PAYS. Pour le type d'article BANQUE/PAYS-R on trouvera 22 articles (renseignements (132 car)) par bloc.
Nous reprendrons ensuite, le même type d'analyse en sachant qu'il y a 3 000 articles BANQUE/PAYS.

a) Volume :

La perte de place est ici plus importante. En effet, 1500 articles BANQUE/PAYS disposent de renseignements (1 sur 2) ce qui donne pour le premier cas, 1 500 x (22 mots) de gaspillés. Dans le deuxième cas, on rajoute 3 000 pointeurs (mots) d'adresses vers les renseignements. Cela donne un avantage de 30 000 mots à la deuxième solution.

b) Temps d'accès :

Au point de vue journalier :

47 accès aléatoires aux articles BANQUE/PAYS sont requis. On n'accède pas à l'item renseignements. L'avantage est ici minime. En effet, ayant accédé à un article BANQUE/PAYS la probabilité de rester dans le même bloc sera de 0.5 % dans le premier cas et de 1.5 % dans le deuxième cas, ce qui est de toute façon négligeable.

Au point de vue bi-annuel :

9 000 accès ; 3 000 demandent l'item renseignements, 6 000 accès (b/p d'un correspondant) sont favorisés par le nombre plus élevé d'articles BANQUE/PAYS dans un bloc. On ne peut déterminer précisément l'avantage d'une solution par rapport à l'autre sans une analyse plus fine . Cependant le temps d'accès est ici moins déterminant.

III COMPTE

Nous reprenons les mêmes hypothèses que pour le type d'article PAYS. On trouve pour la première proposition 16 articles COMPTE par bloc. Pour la deuxième proposition, on trouve 55 articles par bloc et 22 articles COMPTE-R par bloc.

a) Volume :

5 200 comptes dont 2 600 (1 sur 2) disposent de renseignements, ce qui donne 2 600 x 22 mots = 57 200 mots de gaspillés. Dans le deuxième cas, on rajoute 5 200 pointeurs d'adresses vers les renseignements , ce qui donne un avantage de 52 000 mots à la deuxième solution.

b) Temps d'accès :

a) au point de vue journalier :

3 087 accès aléatoires. On n'accède pas à l'item renseignements. La probabilité ayant accédé un compte de retrouver le suivant dans le même bloc sera d'au moins 1 % pour la seconde solution et de 3 pour mille pour la première proposition. Une analyse plus fine du facteur de blocage pourrait être intéressante ici.

b) au point de vue bi-annuel :

10 400 accès aux comptes des correspondants.
Ce type d'accès n'est pas séquentiel. Il ne pouvait l'être que pour une des deux listes. Si l'on charge les correspondants par pays et les comptes par correspondant,

l'accès au compte sera séquentiel dans le cas de la liste des correspondants par banque/pays. Dans le cas où l'on charge les correspondants par banque/pays et les comptes par correspondant, l'accès au compte sera séquentiel pour l'autre liste. Dans le cas de l'accès séquentiel, la deuxième solution est avantageuse. Dans le cas aléatoire , une analyse plus fine est nécessaire mais nous avons dit que le temps d'accès n'était pas un critère décisif pour les applications exceptionnelles.

Similitude avec le schéma conceptuel

La première proposition est bien sûr plus proche du schéma conceptuel car elle en est la traduction. La deuxième proposition n'est cependant qu'une transformation relativement simple de celui-ci.

Facultés offertes aux programmeurs.

Pas de différence marquante entre les deux solutions.
- 1 ordre DDL en plus dans le deuxième cas.

Conclusion

Nous retiendrons donc la seconde proposition qui est satisfaisante pour la majorité des critères et qui est plus performante.

D Propositions d'implémentation pour les types de chemin

- Préalable

Avant de proposer diverses implémentations pour les types de chemin, il nous faut mettre en évidence certaines particularités qui vont restreindre l'éventail de nos propositions.

Analysons d'abord la structure composée des types d'article : VILLE, CORRESPONDANT, BANQUE/PAYS et des types de chemin LOC, EST REPRT.

Nous observons que le type d'article CORRESPONDANT est cible de deux types d'articles VILLE, BANQUE/PAYS respectivement par les types de chemin LOC, EST REPRT. Cette structure n'est donc pas arborescente. Le type d'article CORRESPONDANT ne peut être imbriqué ni dans le type d'article VILLE ni dans celui BANQUE/PAYS auquel cas on ne pourrait respectivement implémenter les types de chemin EST REPRT (BANQUE/PAYS ---> CORRESPONDANT) et LOC (VILLE ---> CORRESPONDANT) et ce à cause de la règle d'existence des subsets manuels et des liens.(cfr DMS - Modèle d'accès)

En conclusion, nous pouvons dire que les types d'articles VILLE, CORRESPONDANT et BANQUE/PAYS doivent être disjoints.

Nous pouvons répéter le même genre de raisonnement avec les types d'article (CORRESPONDANT - ADRESSE SWIFT) et les types de chemin POSS et UTLAS.

En effet, le type d'article ADRESSE SWIFT est cible de deux chemins de types différents issus du type d'article CORRESPONDANT. ADRESSE SWIFT ne peut donc être imbriqué dans CORRESPONDANT par POSS auquel cas on ne pourrait implémenter le type de chemin UTLAS et ce à cause des règles d'existence des subsets manuels et des liens.

Ce même raisonnement doit être appliqué aux types d'article COMPTE, CORRESPONDANT et aux types de chemin PROP, UTLC.

On en déduit que pour les mêmes raisons COMPTE ne peut être imbriqué dans CORRESPONDANT.

En ce qui concerne les temps d'accès , nous discuterons ceux-ci en prenant le nombre moyen d'articles cibles d'un type de chemin comme facteur de blocage. On améliorera cependant cette proposition lors d'un réglage plus fin des paramètres physiques. On ne peut, en effet, ici fixer le facteur de blocage définitif étant donné que nous avons aucune information sur la distribution du nombre d'articles cibles par article origine. Nous ne possédons que la moyenne. Cette hypothèse nous suffira néanmoins pour discriminer, dans un premier temps, deux propositions.

Type de chemin CONT (PAYS ---> VILLE)

a) Caractéristiques

Cardinalité (1-N), fort pour ville ; type de chemin utilisé pour la liste des correspondants.
Accès séquentiel aux villes d'un pays (1 000 tous les six mois)

b) Propositions

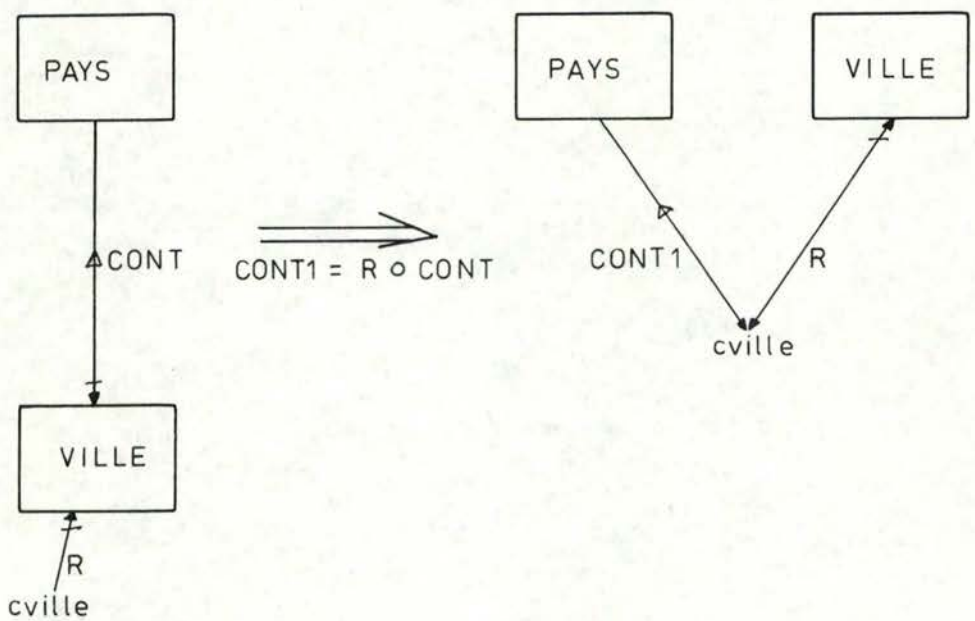
1) Imbrication du type d'article VILLE dans le type PAYS.

2) Subset manuel (M-N) avec contraintes

3) Lien (M-N) avec contraintes

4) Transformation de structure

Migration de type de chemin.
La cible devient un item identifiant du type d'article cible.



avec contrainte : toute valeur du code ville doit appartenir à un et un seul pays.

c) Analyse

Dans le préalable , nous avons signalé que le type d'article VILLE devait être de niveau disjoint.Cela implique que nous ne pouvons ici , employer la première proposition. Analysons donc les trois autres suivant les critères énoncés.

1) Porte ouverte aux extensions

Il n'y a ici , aucune différence entre les trois solutions, étant donné le fait que les deux types d'article (cible et origine) se trouvent au niveau disjoint. Le type de chemin inverse peut être facilement implémenté soit par un lien (N-1), soit par une transformation similaire à celle de la quatrième proposition. Si on demande une clé d'accès sur le chemin , celle-ci ne pourra être fournie qu'avec la proposition du subset manuel.

2) Facilités offertes aux programmeurs

Les trois propositions restantes n'offrent aucune facilité au programmeur. L'insertion est manuelle et la vérification des contraintes est à sa charge. De plus il existe un problème de définition du nombre maximum de villes par pays dans les troisième et quatrième cas. Le nombre maximum d'articles cibles d'un chemin doit être fixé lors de la définition du schéma (clause OCCURS). Cela influencera, nous le verrons, les performances (volumes). La contrainte imposée par la transformation est à charge du programmeur.

3) Performances

Nous avons vu que nous prenions le nombre moyen d'articles cibles comme facteur de blocage soit ici, 5 articles VILLE dans un bloc.

a) Volumes
---

1) Subset manuel

Comme vu en annexe (Précisions sur l'organisation des data sets, sets et subsets), nous trouverons en moyenne six mots d'adresses par chemin (5 pour les villes villes d'un pays dans le subset + 1 vers cette table du subset).

200 x 6 mots = 2 200 mots d'adresses.

2) Lien

Le nombre maximum des articles cibles des chemins de ce type doit être fixé à 20. Nous avons donc 200 x 20 ==> 4 000 mots d'adresses.

3) Transformation de structure

Le code ville traduit le chemin d'accès. Il doit être déclaré répétitif (OCCURS) dans le schéma. Nous avons donc 200 x 3 car x 20 = 2 000 mots. De plus, il nous faut définir une clé d'accès sur le type d'article (cville) ce qui coûte 1 000 x 3 car + 1 000 mots d'adresses soit 1 500 mots et un total de 3 500 mots.
La première solution est donc la plus avantageuse du point de vue du volume

b) Temps d'accès
---

1) Subset manuel

Dans une table, on ne retrouve que les adresses vers les articles cibles d'un même chemin. Fixons donc, la taille des tables à 5. Nous ne pouvons déterminer exactement dans combien de cas, la lecture d'une table suffira à obtenir les adresses des articles cibles mais une lecture sera de toute façon nécessaire. Dans le cas le plus défavorable, quatre lectures seront nécessaires. De une à quatre lectures physiques de bloc seront également nécessaires. On pourrait donc penser à modifier le nombre d'entrées par table ce qui diminuerait l'avantage de la solution au point de vue volume.

2) Lien

Nous avons fixé le facteur de blocage du type d'article VILLE à 5. Nous avons donc de 1 à 4 lectures physiques de blocs suivant la distribution des villes par pays.

3) Transformation de structure

Ce raisonnement est identique à celui du subset manuel car l'accès se fait par l'intermédiaire d'un set avec le désavantage que les codes des villes d'un pays ne sont pas nécessairement séquentiels. On peut cependant ici augmenter le nombre d'entrées par table afin d'augmenter la probabilité des rester la dernière table lue. Ceci mériterait une analyse plus fine.

4 Clareté

Les deux premières solutions (2,3) sont une traduction approximative du schéma conceptuel. En effet, les contraintes ne sont pas prises en compte par ces solutions et il faudra veiller à modifier la cardinalité supportée par ce mécanisme (M-N). La dernière est une transformation élémentaire du schéma où les contraintes ne sont pas non plus prises en compte.

Conclusion

Nous retiendrons la solution du subset manuel qui est favorable pour la majorité des critères, excepté le temps d'accès. Etant donné le caractère extraordinaire (tous les six mois durant la nuit) de l'application employant ce type de chemin et la différence minime de temps par rapport aux autres solutions, le temps d'accès ne peut être discriminant.

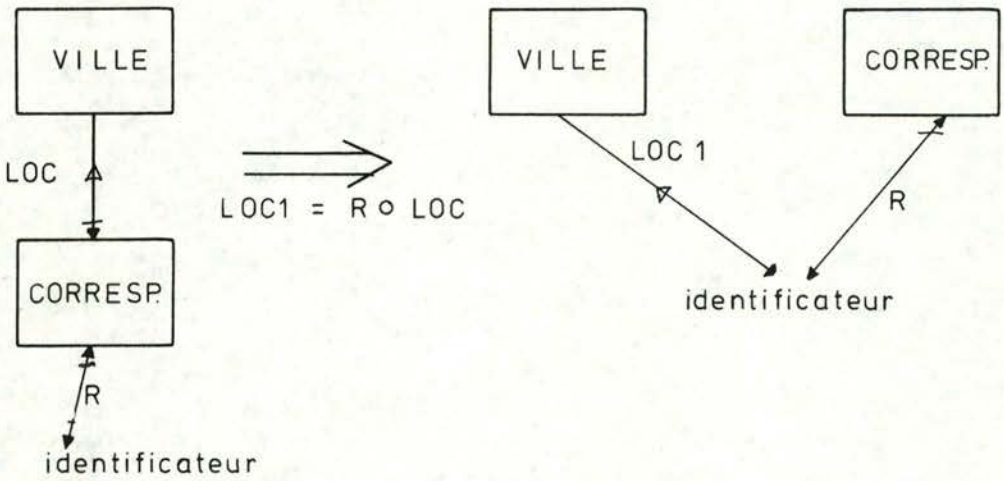
Type de chemin LOC (VILLE ---> CORRESPONDANT)

a) Caractéristiques

Cardinalité (1-N), fort pour correspondant.
Type de chemin utilisé pour la liste des correspondants
Accès séquentiel aux correspondants d'une ville (6 000 tous les six mois)

b) Proposition

cfr CONT (1, 2, 3)
La quatrième proposition étant une transformation similaire



c) Analyse

On reprendra l'analyse effectuée pour le type de chemin (CONT) en modifiant les facteurs de blocage et la taille des tables qui prennent la valeur 6.

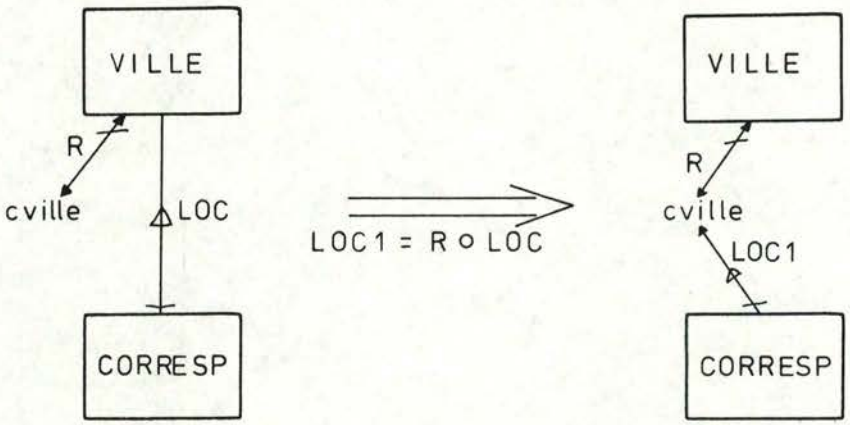
Type de chemin LOC (CORRESPONDANT ---> VILLE)

a) Caractéristiques

Cardinalité (N-1) fort pour correspondant.
47 fois par jour + 6 000 tous les six mois.

b) Propositions

1) Lien (N-1)
2) Transformation de structure



Il faut définir une clé d'accès sur VILLE (code ville)

c) Analyse

1) Porte ouverte aux extensions

Pas de différence

2) Facilités offertes aux programmeurs

Les deux solutions ne permettent que l'insertion manuelle à savoir dans le premier cas, établir le lien (ASSIGN) et dans le deuxième cas, garnir le code ville du correspondant. La deuxième solution permet de rendre le chemin obligatoire pour correspondant en définissant l'item code ville obligatoire (REQUIRED). La première solution permet d'empêcher qu'une ville comprenant encore un correspondant puisse être supprimée (COUNTED LINK).
Au point de vue des ordres DDl, dans le premier cas, un ordre explicite prévu, dans le deuxième cas, on utilisera le SET défini sur VILLE avec comme clé d'accès identifiante code ville.

3) Performances

a) Volume

Dans la première solution , nous aurons un pointeur (6 car) par correspondant vers sa ville + un compteur (2 car) par ville qui contiendra le nombre d'articles CORRESPONDANT pointant vers celle-ci
===> 6 x 6 000 + 2 x 1 000 = 38 000 car soit 6 333 mots

Dans la deuxième solution nous aurons un code ville (3 car.) par correspondant + un set défini sur le type d'article VILLE ===> un pointeur d'adresse (6 car.) + un code ville (3 car.) par article VILLE
6 000 x 3 car . + 1 000 x 9 car. = 27 000 car soit 4 500 mots.

L'avantage est donc à la deuxième proposition qui offre en plus une clé d'accès sur le type d'article ville

b) Temps d'accès

La première solution sera plus avantageuse car un seul accès suffira toujours pour trouver la ville d'un correspondant.
Dans le deuxième cas, il faudra parfois lire la table des adresses par code ville (SET). Pour réduire ce désavantage, on pourrait ici considérer une taille de table importante (340 entrées) ce qui équivaut à 510 mots (9 car. = 1.5 mots)
Cette solution porte à au moins 32 % la probabilité de ne pas devoir lire une autre table dans le cas d'accès aléatoire (quotidien) ce qui n'est pas négligeable.

4 Clareté

La première solution reflète exactement le schéma conceptuel à condition de respecter les contraintes par programmes. La deuxième est une transformation élémentaire de celui-ci.

5 Conclusion

Nous retiendrons cette dernière car elle nous offre en plus une clé d'accès à code ville tout en économisant de la place bien que le temps d'accès soit un peu plus défavorable. On accède aux VILLES seulement 47 fois par jour par ce chemin.

Type de chemin COMP (BANQUE/PAYS ---> PAYS)

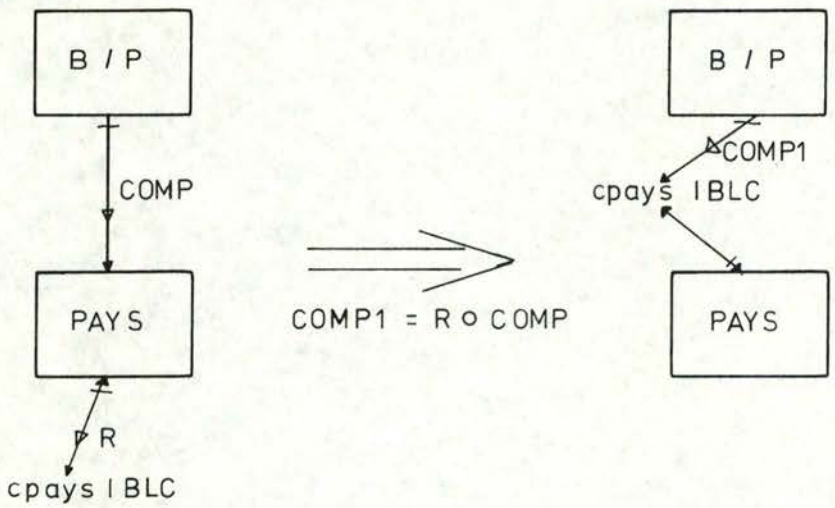
a) Caractéristiques

Cardinalité (N-1), fort pour BANQUE/PAYS
Accès pour la liste des correspondants (3 000 fois tous les six mois)

b) Proposition

1) Lien (N-1)

2) Transformation de structure



c) Analyse

cfr LOC (CORRESPONDANT ---> VILLE)
Solution identique avec l'avantage que le set sur PAYS est déjà implémenté et qu'il n'y a plus d'accès quotidien.

Type de chemin REPR (BANQUE ---> BANQUE/PAYS)

a) Caractéristiques

Cardinalité (1-N) fort pour BANQUE/PAYS.
Accès séquentiel aux articles BANQUE/PAYS d'une banque.
Liste des correspondants (3 000 tous les six mois)

b) Propositions

cfr CONT (1,2,3) Nous n'envisageons pas ici la quatrième solution car il faudrait définir une clé identifiante sur BANQUE/PAYS (cbque, cpays IBLC) qui accroîtrait les désavantages de la proposition.

c) Analyse

Nous retiendrons poour les mêmes raisons que celles présentées lors de l'analyse de CONT la solution du subset manuel.

Type de chemin EST REPRT (BANQUE/PAYS --->CORRESPONDANT)

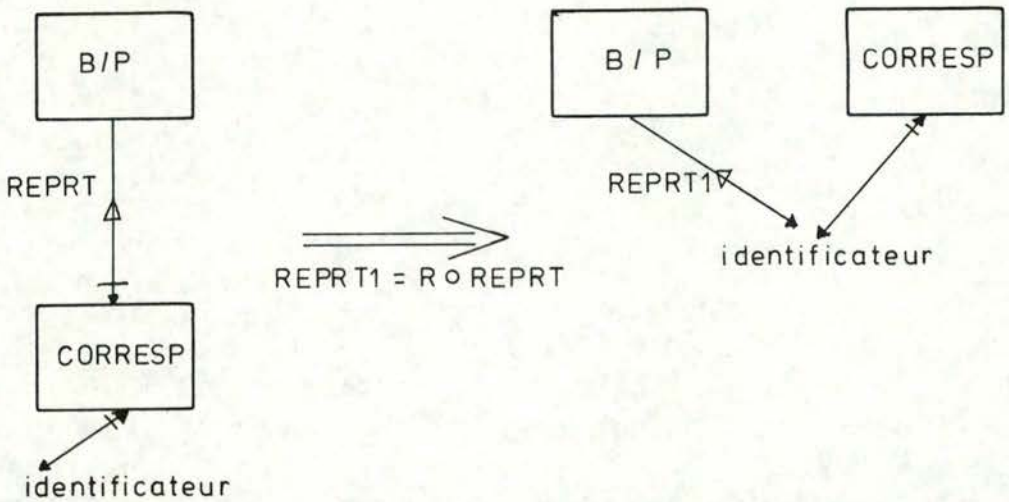
a) Caractéristiques

Cardinalité (1-N) fort pour correspondant.
Accès séquentiel aux articles correspondants d'une banque/pays

b) Proposition

cfr CONT (1,2,3)

4,) Transformation de structure



avec contrainte : toute valeur d'identificateur doit appartenir à un et un seul article BANQUE/PAYS

c) Analyse :

Les caractéristiques étant identiques , nous pouvons reprendre l'analyse du type de chemin CONT et proposer un subset manuel pour implémenter ce type de chemin.

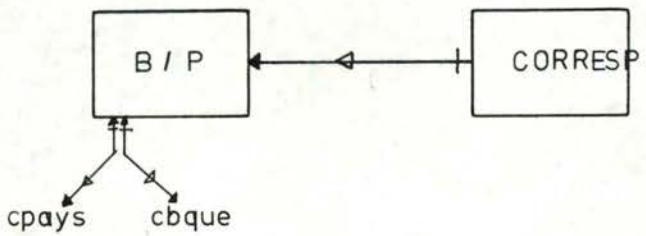
Type de chemin EST REPRT (CORRESPONDANT ---> BANQUE/PAYS)

a) Caractéristiques :

Cardinalité (N-1) ; fort pour correspondant.
Accès à la banque/pays d'un correspondant (6 000 fois tous les six mois) + 47 par jour.

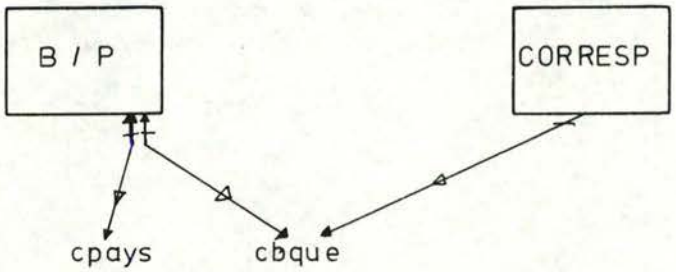
b) Propositions

1) Lien (N-1)



2)

En rajoutant les items code banque et code pays qui forment un identifiant de banque/pays, on peut proposer la transformation de structure déjà présentée



Cette solution est cependant moins intéressante. Il faut en effet rajouter ces deux items (7 car.) aux articles du type banque/pays (3000). Il faut de plus définir un SET sur banque/pays avec ces deux items comme clé identifiante. Enfin, le temps d'accès est défavorable et nous avons des accès journaliers. En conséquence , nous retiendrons la première proposition.

Type de chemin UTLAS (CORRESPONDANT ---> ADRESSE SWIFT)

a) Caractéristiques :

Cardinalité (M-N) avec clé d'accès cserv (304 accès par jour)

b) Proposition :

Subset manuel (M-N) avec clé d'accès cserv.

c) Analyse :

Nous n'analyserons pas ce type de chemin , la proposition étant unique et le subset manuel étant , nous l'avons vu, une bonne solution.

Type de chemin POSS (ADRESSE SWIFT ---> CORRESPONDANT)

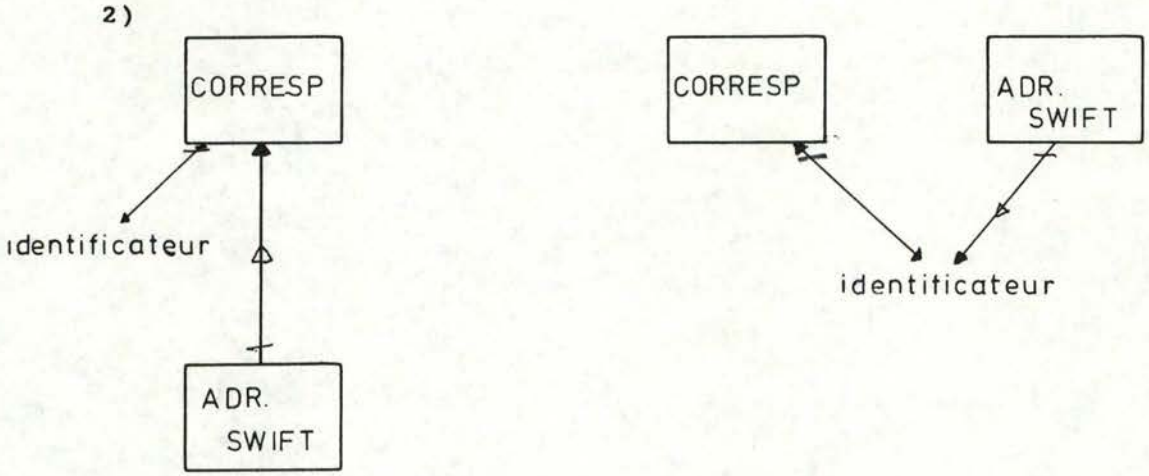
a) Caractéristiques

Cardinalité (N-1) fort pour adresse swift.
1 520 accès aléatoires par jour.

b) Proposition
---

1) Lien (N-1)

2)



c) Analyse

Ici, le temps d'accès devient une contrainte non négligeable. Pour les raisons citées plus haut (analyse de LOC) concernant celui-ci , nous retiendrons donc la solution du lien , bien que le set soit déjà défini sur CORRESPONDANT et que le code banque et le code pays se trouvent déjà dans l'adresse swift.

Type de chemin POSS (CORRESPONDANT ---> ADRESSE SWIFT)
---

a) Caractéristiques

Cardinalité (1-N) fort pour adresse swift
1 064 accès aléatoires par jour.

b) Propositions
---

1) Imbrication avec un set défini sur le type d'article
Adresse swift : cserv

2) Subset manuel avec clé (cserv)

c) Analyse

Nous devons tout de suite écarter la première solution étant donné les raisons exposées dans le préalable. La deuxième proposition s'impose donc.

Type de chemin DISP (CORRESPONDANT ---> ADRESSE TELEX)
---

a) Caractéristiques

Cardinalité (1-N) ; fort pour adresse télex avec clé d'accès cserv.
162 accès aléatoires par jour.

b) Proposition
---

1) Imbrication du type d'article ADRESSE TELEX d'organisation ORDERED (ACCESS sur CSERV) ou avec set défini sur cserv.

2) Subset manuel avec clé d'accès cserv.

c) Analyse
---

1) Possibilité d'extension

Ici, nous écarterons l'organisation ORDERED du type d'article ADRESSE TELEX car celle-ci empêcherait la définition d'une autre clé d'accès aux adresses télex.

2) Facilités offertes aux programmeurs

Toutes les contraintes sont à charge du système dans le premier cas , alors que leur respect est à charge du programmeur dna le deuxième (fort pour ADRESSE TELEX, 1-N)

3) Performances

a) Volume

- Première proposition

Par article maître, un pointeur + X pointeurs vers ses articles cibles soit X + 1 pointeurs.

-Deuxième proposition

Par article maître, un pointeur vers la table des articles cibles + X pointeurs vers ceux-ci + X occurrence du code service (2 car.)

La deuxième proposition est un peu plus défavorable.

b) Temps d'accès

La deuxième proposition entraînera souvent la lecture d'une table des articles cibles que la deuxième proposition évite. Cependant, la recherche de l'article cible sera plus rapide dans le deuxième cas. En effet, il faudra lire la chaîne des articles cibles et examiner leur code service. on peut difficilement, faute d'informations chiffrer, ici, la différence.

4) Clareté

Pas de différence entre les deux solutions.

5) Conclusion

Etant donnè la facilitè qu'offre la première solution, nous la retiendrons. Pour les raisons exprimées lors de l'étude des possibilitès d'extension, nous retiendrons la première solution avec un SET défini sur ADRESSE TELEX.

Type de chemin ADRP (CORRESPONDANT ----> ADRESSE POSTALE)

a) Caractèristiques :

Cardinalitè (1-N), fort pour ADRESSE POSTALE
Type de chemin avec clè d'accès (cserv)
42 accès alèatoires par jour.

b) Propositions

cfr DISP

c) Analyse

cfr DISP

Type de chemin ADRT (CORRESPONDANT ---> ADRESSE TELEGRAPHIQUE)

a) Caractèristiques

Cardinalitè (1-N), fort pour adresse tèlègraphique
Type de chemin avec clè d'accès (cserv)
28 accès alèatoires par jour

b) Propositions

cfr DISP

c) Analyse

cfr DISP

Type de chemin PROP (CORRESPONDANT---> COMPTE)

a) Caractèristiques

Cardinalitè (1-N) fort pour compte
Type de chemin avec clè d'accès (cserv)
456 accès alèatoires par jour.

b) Propositions

a) Imbrication du type COMPTE dans CORRSPONDANT avec un set qui dèfinit la clè : cserv.

b) Subset manuel (M-N) + contraintes avec clè d'accès sur cserv.

c) Analyse

La première proposition est à èliminer pour les raisons exposèes dans le prèalable. Il nous reste donc la deuxième solution du subset manuel.

Type de chemin PROP (COMPTE ---> CORRESPONDANT)

a) Caractèristiques

Cardinalitè (N-1) fort pour COMPTE
2631 accès alèatoires par jour.

b) Propositions

a) Lien (N-1)

b) Transformation de structure

c) Analyse

Ici, la contrainte du temps d'accès prime sur les autres critères . Ce type de chemin est en effet, le plus utilisé. Aussi, nous retiendrons sans analyse plus élaborée , la solution du lieu étant donné son temps d'accès plus efficient. (cfr LOC (CORRESPONDANT ---> VILLE)).

Type de chemin UTLC (CORRESPONDANT ---> COMPTE)
---

a) Caractéristiques

Cardinalité (M-N)
1 064 accès aléatoires par jour .

b) Proposition
---

a) Subset manuel (M-N) avec clé d'accès (cserv)

c) Analyse

Cette proposition a l'avantage d'être simple et intéressante au point de vue performance (cfr avant). On pourrait proposer une transformation de structure qui n'améliorerait pas les performances (il faudrait de toute façon passer par un ou plusieurs sets) et rendrait la structure plus complexe.

Type de chemin DEP (COMPTE ---> CORRESPONDANT)
---

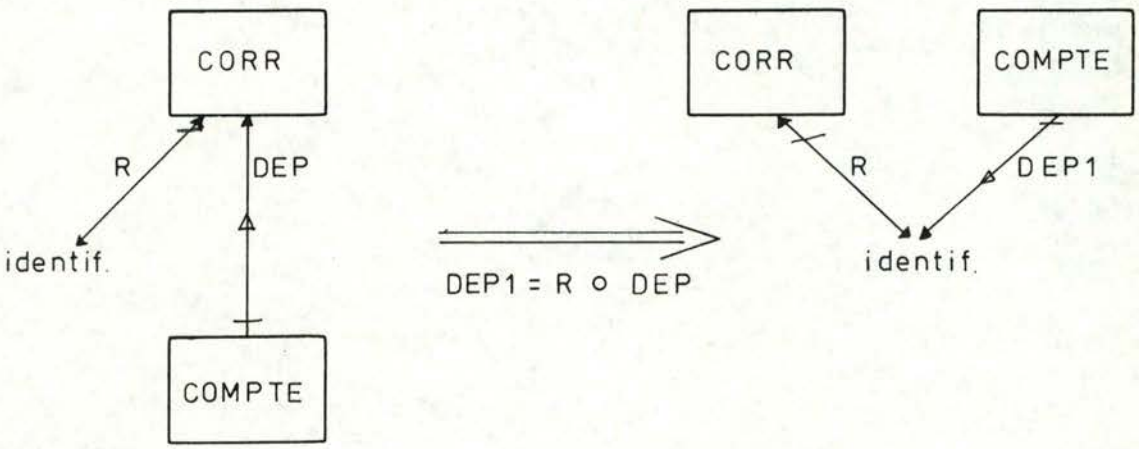
a) Caractéristiques

Cardinalité (N-1) , fort pour COMPTE
1 520 accès aléatoires par jour.

b) Proposition
---

a) Lien (N-1)

b) Transformation de structure



c) Analyse

La contrainte de temps d'accès joue ici, de nouveau en faveur de la première proposition que nous retiendrons.

1) Type de chemin HISTO (COMPTE ---> ANCIEN COMPTE)
2) Type de chemin ANC/NOU (ADRESSE SWIFT ---> ANCIENNE ADRESSE SWIFT)

a) Caractéristiques

Cardinalité (m-N) fort pour ANCIEN COMPTE (1) et ANCIENNE ADRESSE SWIFT (2).
Emploi non spécifié

b) Proposition

a) Subset manuel (M-N)

b) Lien (OCCURS) (M-N)

c) Analyse

Etant donné la difficulté de fixer un nombre maximum d'articles

cibles pour un article origine pour la deuxième proposition et la perte de place qu'entraînerait ce lien (peu d'occurrence de ce chemin alors que la place serait réservée pour tout article potentiellement origine) , nous retiendrons la première proposition.

E. Volume estimé

Nous avons estimé le volume physique de la base (3 000 000 car). On se rapportera à l'annexe 5 pour obtenir des informations plus précises sur cette estimation.

CONCLUSION

CONCLUSION

Nous avons, dans cet ouvrage, tenté de présenter la conception de la base de données d'un système d'information en présentant la démarche qui a guidé notre analyse.

Nous n'avons pas la prétention d'affirmer que la démarche que nous avons suivie était la bonne. Nous voudrions terminer cette étude en examinant d'autres méthodes telles celle proposée par H.TARDIEU, NANCI, et PASCOT dans l'ouvrage "Conception d'un système d'information" ou celle définie par G.BENCI et C.ROLLAND dans "Base de données". On retrouve dans ces méthodes la décomposition en niveaux qui a été mise en évidence à NAMUR dès 1974 dans "Data structure models for information systems" et en 1975 par le rapport ANSI/SPARC. Les niveaux que nous avons présentés se retrouvent dans ces méthodes sous des noms différents. On peut d'ailleurs reprendre ici, les niveaux communs.

La première étape est une représentation du réel perçu dans un modèle appelé conceptuel, indépendant de toute contrainte informatique. La différence essentielle entre les méthodes est, à ce niveau, le modèle employé pour décrire le système. Tardieu propose un modèle élaboré par son équipe : le modèle individuel. Benci propose, lui, le modèle de CODD. Il voit d'ailleurs un avantage, bien que ce ne soit pas la raison de son choix, à ce modèle : le court-circuitage des étapes suivantes lors de l'apparition de SGBD de type relationnel qui n'existe encore qu'à l'état de prototypes. Nous avons, nous , proposé un modèle ENTITE/RELATION qui est en fait un sous-ensemble (limité aux associations binaires) du modèle ENTITE/ASSOCIATION dont les concepts essentiels ont été notamment mis en évidence par Messieurs Bodart, Benci, Cabanes, Bogaert et Chen.

Le deuxième niveau que l'on retrouve dans ces méthodes est la transformation de la représentation conceptuelle dans un modèle d'accès logiques aux données. Le modèle proposé par Benci est inspiré des propositions de Cabannes dans "Un modèle d'accès standard". Le modèle que nous avons employé, développé par J-L Hainaut ainsi que celui proposé par Tardieu est, lui, basé sur les propositions CODASYL.

Le troisième niveau qui est une étude des besoins par l'analyse des applications permet de déterminer le schéma des accès standards (Tardieu) ou nécessaires (Namur). Ce troisième niveau qui est peu développé chez Benci est très semblable à ce que Namur propose chez Tardieu.

Le dernier niveau, enfin, qui est une traduction du schéma obtenu à l'étape précédente dans le langage de description (DDL) d'un SGBD en utilisant une correspondance entre les concepts du modèle d'accès logique et leur traduction dans le SGBD. Tardieu n'aborde pas cette correspondance dans son travail.

Comme nous venons de le voir plus haut , les méthodes citées ci-dessus diffèrent essentiellement par les outils méthodologiques mis en oeuvre à chaque étape et ceux utilisés pour passer d'une étape à l'autre. C'est d'ailleurs à cette hiérarchie de niveaux et plus précisément au précepte qui dit qu'un niveau est indépendant des critères pris en charge aux niveaux inférieurs, que nous avons des commentaires à apporter.

En effet, et en tout cas dans le développement de notre système , le processus d'analyse ne fut pas purement séquentiel. Il nous a fallu à plusieurs reprises, revenir à l'analyse conceptuelle alors que nous étions déjà à une étape ultérieure. Cela était dû , soit à une non compréhension de notre part, soit parce que , depuis le début de l'analyse , des modifications étaient intervenues. S'il est possible de remédier à la non compréhension, soit en mettant à la disposition du responsable du système les outils nécessaires à la description conceptuelle du système , soit en insérant un membre de la cellule de conception dans le groupe responsable du système , il nous paraît difficile d'apporter une solution au deuxième problème dû à une évolution rapide d'un système en cours de développement. Pour ce genre de système où aucune connaissance stable n'existe encore, nous aurions tendance à préférer une démarche intuitive. Cette approche serait à mettre en relation avec le développement de maquettes, outils permettant de simuler une réalisation du système à moindre frais bien que ce développement soit loin d'être évident.

Nous voudrions, enfin aborder ce que beaucoup reproche à ce genre de méthode : le temps que son application requiert. On peut émettre bien des avis sur ceci et proposer de réduire certaines étapes. Il nous semble important néanmoins de ne pas négliger la phase d'analyse conceptuelle, car et nous en avons eu l'expérience beaucoup de retours en arrière sont nécessaires à cause d'une mauvaise définition des concepts de base du système. Il faut également savoir qu'un certain nombre d'étapes, par exemple, la caractérisation du SGBD ne seront effectuées qu'une fois et serviront lors de toutes les analyses futures. Enfin, l'expérience acquise du SGBD réduira fortement le temps de la dernière étape.

Nous sommes persuadés que les dernières remarques que nous avons apportées mériteraient de plus amples développements . Elles constituent , nous semble-t-il , une excellente piste de réflexion.

BIBLIOGRAPHIE

BIBLIOGRAPHIE.

[1] HAINAUT, J-L. : "Conception de la base de données d'un système d'information", Institut d'Informatique, 1981.

[2] DELVAUX,Y., Hainaut,J-L. : "Système portable de manipulation de bases de données hétérogènes", Actes Congrès AFCET 1981.

[3] HAINAUT,J-L. : "Theoretical and Practical Tools for Data Base Design",Proc. VLDB 1981 (ACM).

[4] BENCI,G., BODART,F., BOGAERT, H., CABANES, A. : "Concepts for the Design of a Conceptual Schema" , in "Modelling in Data Base Mangement Systems", North-Holland , 1976.

[5] CHEN, P. : "The Entity-Relationship Model : Toward a Unified View of Data", ACM TODS 1, 1, 1976

[6] DATE : "An Introduction to Data Base System", Addison-Wesley, 1977

[7] HAINAUT, J-L. : "Construction de schèmas conceptuels d'une base de données : étude de cas numéro 1 " Institut d'Informatique, Namur, 1980

[8] HAINAUT,J-L. : "Dérivation d'une première structure d'accès à partir d'un schéma conceptuel Entité/Association", Institut d'Informatique, Namur , 1980

[9] TARDIEU, H., NANCI, PASCOT : "Conception d'un système d'Information", Les éditions d'organisation / Gaetan Morin, 1979.

[10] BENCI,G., ROLLAND,C. :"Bases de données", ed. SCM, Paris, 1979

[11] HAINAUT,J-L. : "Modèles Conceptuels", Lecture notes, Public. Institut d'Informatique, Namur , 1980

[12] HAINAUT,J-L :"Un modèle de description de fichiers : le modèle d'accès",Lecture notes, Public. Institut d'Informatique, Namur, 1980

[13] BURROUGHS B 7000 / B 6000 SERIES
DMS II DASDL
Reference Manual , Mars 1978

[14] BURROUGHS B 7000 / B 6000
DMS II HOST
Reference Manual, Juin 1977

[15] VAN LAMSWEERDE ,A. : "Cours d'Analyse Organique", Notes, 1980.

[16] BODART,F. : "Analyse Conceptuelle - Fonctionnelle "
Notes de cours, 1980.

[17] "Data Structure Models for Information Systems",
"Proc. Intern. Workshop of Namur, 1974",
Presses Universitaires de Namur, 1975.

[18] YON, D. : "Etude comparative sur les systèmes de gestion de Base de Données", Burroughs, 1979.

ANNEXES

Annexe 1 :

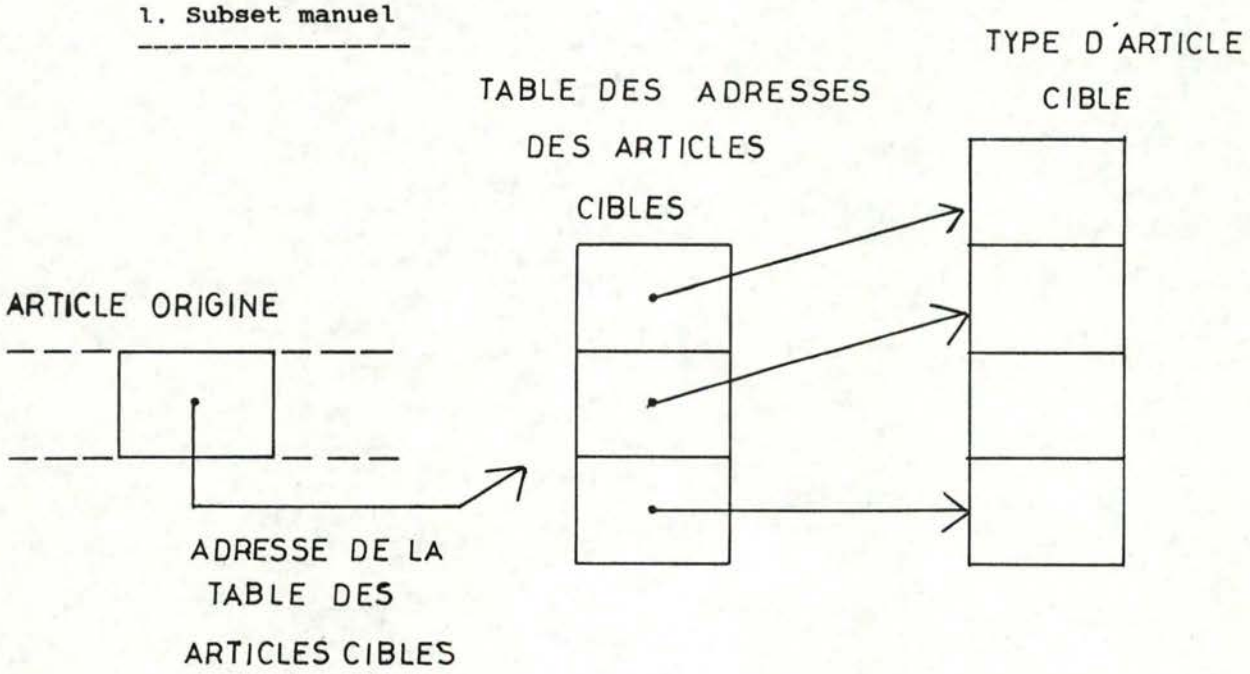
Précisions sur l'organisation physique des data sets, sets, et subsets

a) Bloc - table - buffers

Les data sets sont organisés en bloc. Les sets et subsets en table. L'unité de transfert de mémoire secondaire vers la mémoire principale est dans le premier cas le bloc , dans le second cas, la table. La longueur de ces unités est spécifiable dans la description du schéma par les clauses BLOCKSILE, TABLESILE. Cette longueur peut être exprimée en mots,segments, articles.
Par data set, set, subset on peut allouer un certain nombre de buffers (espace mémoire pouvant contenir, dans le cas d'un data set un bloc, dans le cas d'un subset ou d'un set une table). Ce nombre est spécifiable par la clause BUFFERS associée à chaque data set, set, subset. Dans la plupart des cas, en effet, en augmentant le nombre de buffers, on aura tendance à diminuer le nombre d'échange entre mémoire secondaire et mémoire principale, et donc à réduire le temps d'accès aux articles.

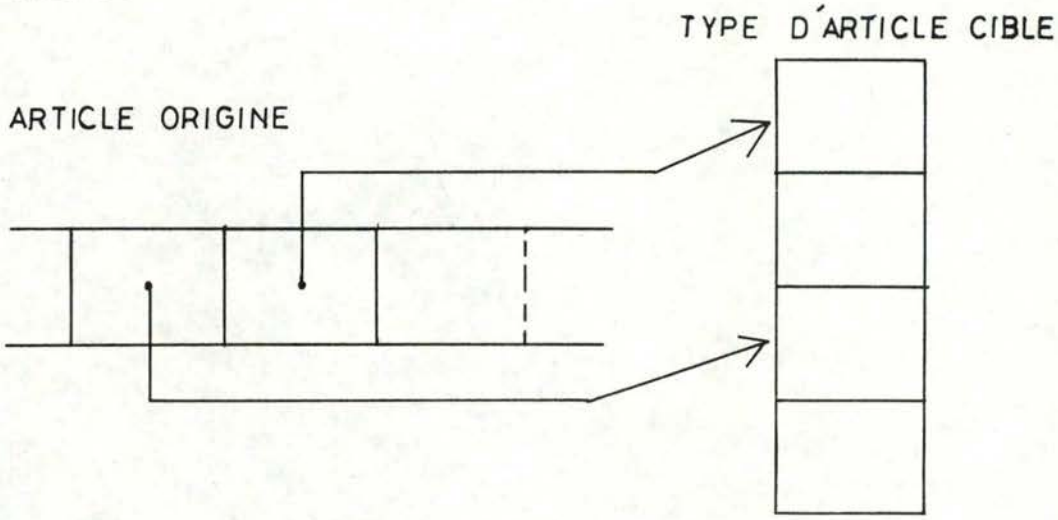
b) Organisation des types de chemin

1. Subset manuel



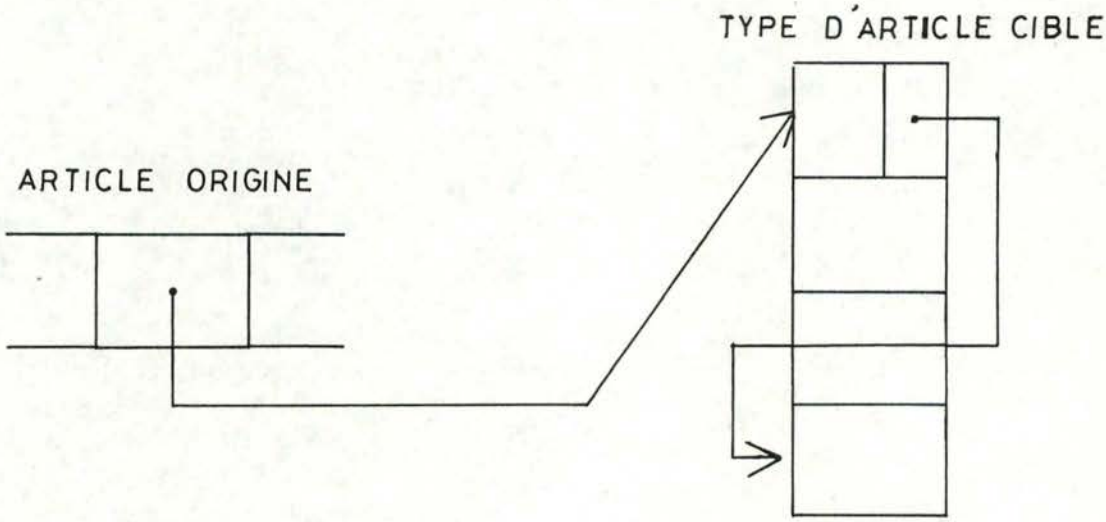
Dans l'article origine du chemin, se trouve un pointeur (48 bits) vers la table des adresses des articles cibles du chemin. Dans ce cas, nous avons donc n+1 pointeurs par chemin , n étant le nombre d'articles cibles de ce chemin. Les tables ne peuvent contenir que les adresses des articles d'un même chemin.

2. Lien



Dans l'article origine , se trouve 1 ou N (OCCURS) pointeurs vers les articles cibles de ce chemin. Nous avons donc ici 1 ou N pointeurs par chemin , N étant une constante définie dans la description du schéma comme le nombre maximum d'articles cibles d'un chemin de ce type.

3. Data set imbriqué



Dans l'article origine se trouve un pointeur vers un article cible du chemin. Les autres articles sont chaînés entre eux à partir de celui-ci. Suivant le type d'organisation , il se trouve dans le même bloc ou non

Annexe 2 : Schéma DDL de la base "correspondants".

```
PAYS DATA SET

(clangue              NUMBER (2)  OCCURS  3 TIMES;
cdev                  NUMBER (3);
nom-français          ALPHA (20);
nom-néerlandais       ALPHA (20);
cpays-RTT             ALPHA (3);
cpays-INS             NUMBER (3);
cpays-ISO             ALPHA (2);
cpays-IBLC            NUMBER (3)  REQUIRED;
limite-risque         NUMBER (15);
renseignements-p      REFERENCE TO   PAYS-R  COUNTED;
access-ville          SUBSET OF VILLE;
);

access-pays-INS SET OF PAYS
KEY IS cpays-INS
DUPLICATES;
access-pays-IBLC SET OF PAYS
KEY IS  cpays-IBLC
NO DUPLICATES;

VILLE DATA SET

(cville               ALPHA (3)  REQUIRED;
nom                   ALPHA (20);
access-corresp SUBSET OF CORRESPONDANT;
);

Access-ville SET OF VILLE
KEY IS cville
NO DUPLICATES;
```

PAYS-R DATA SET

```
(renseignements      ALPHA (132)  REQUIRED;
count                NUMBER (2)   REQUIRED;
);
```

BANQUE-PAYS-R DATA SET

```
(renseignements      ALPHA (132)  REQUIRED;
count                NUMBER (2)   REQUIRED;
);
```

BANQUE-PAYS DATA SET

```
(identif -siège-principal    ALPHA (10);
raison-sociale               ALPHA (60);
renseignements-b-p  REFERENCE TO BANQUE-PAYS-R COUNTED;
cpays-IBLC          NUMBER (3)  REQUIRED;
access-correspondant  SUBSET OF CORRESPONDANT;
);
```

BANQUE DATA SET

```
(cbanque             ALPHA (4)  REQUIRED;
raison-sociale       ALPHA (60);
cpays-origine        NUMBER (3);
limite-risque        NUMBER (15);
access-banque-pays   SUBSET OF BANQUE-PAYS;
);
```

CORRESPONDANT DATA SET

```
(identificateur      ALPHA (10)  REQUIRED;
raison-sociale       ALPHA (15);
renseignements       FIELD (4);
cville               ALPHA (3)  REQUIRED;
access-banque-pays   REFERENCE TO BANQUE-PAYS COUNTED;
access-adresse-swift-utl  SUBSET OF ADRESSE-SWIFT KEY IS cserv;
```

```
access-adresse-swift-poss SUBSET OF ADRESSE-SWIFT KEY IS cserv;
access-compte-prop  SUBSET OF COMPTE KEY IS cserv;
access-compte-utlc  SUBSET OF COMPTE KEY IS cserv;


ADRESSE-TELEX DATA SET


(identificateur     ALPHA (10) REQUIRED;
cserv               FIELD (16) REQUIRED;
échange-de-clé      FIELD;
);
access-adresse-télex  SET OF ADRESSE-TELEX KEY IS cserv;


ADRESSE-POSTALE DATA SET


(intitule           ALPHA (30)  REQUIRED;
échange-signature   FIELD;
cserv               FIELD (16)  REQUIRED;
);
access-adresse-postale  SET OF ADRESSE POSTALE KEY IS cserv;


ADRESSE TELEGRAPHIQUE


(identificateur     ALPHA (20)  REQUIRED;
cserv               FIELD (16)  REQUIRED;
);
access-adresse-télégraphique  SET  OF  ADRESSE-TELEGRAPHIQUE  KEY   IS
cserv;


COMPTE DATA SET


(numéro             NUMBER (12)  REQUIRED;
cdev                NUMBER (3)   REQUIRED;
cserv               FIELD (16)   REQUIRED;
renseignements-c  REFERENCE TO COMPTE-R COUNTED;
access-correspondant-prop  REFERENCE TO CORRESPONDANT COUNTED;
access-correspondant-dep  REFERENCE TO CORRESPONDANT COUNTED;
access-ancien-compte  SUBSET OF COMPTE;
);
```

```
ANCIENNE-ADRESSE-SWIFT


(identificateur      ALPHA (10)  REQUIRED;
data-de-maj          NUMBER (8)  REQUIRED;
access-adresse-swift  SUBSET OF ADRESSE-SWIFT;
);
access-ancienne-adresse-swift SET OF ANCIENNE-ADRESSE-SWIFT
KEY is identificateur
NO DUPLICATES;


ANCIEN-COMPTE


(numéro              NUMBER (12)  REQUIRED;
cdev                 NUMBER (3)   REQUIRED;
data-de-maj          NUMBER (8)   REQUIRED;
access-compte  SUBSET OF COMPTE;
);
access-ancien-compte SET OF ANCIEN COMPTE
KEY IS numéro , cedv
NO DUPLICATES;


COMPTE-R  DATA SET


(renseignements      ALPHA 9132)  REQUIRED;
count                NUMBER (2)   REQUIRED;
);


ADRESSE-SWIFT DATA SET


(identificateur      ALPHA (11)  REQUIRED
authenticateur       FIELD;
cactive              FIELD;
cserv                FIELD (16)  REQUIRED;
data-dern-modif      NUMBER (8);
access-ancienne-adresse  SUBSET OF ANCIENNE ADRESSE SWIFT;
access-correspondant  REFERENCE TO CORRESPONDANT COUNTED;
);
```

Annexe 3 : Image des items par types d'article

PAYS :

clangue	NUMBER (2)
cdev	NUMBER (3)
nom français	ALPHA (20)
nom néerlandais	ALPHA (20)
cpays RTT	ALPHA (3)
cpays INS	NUMBER (3)
cpays ISO	ALPHA (2)
cpays IBLC	NUMBER (3)
renseignements	ALPHA (132)

BANQUE :

cbanque	ALPHA (4)
raison sociale	ALPHA (60)
cpays origine	NUMBER (3)
limite risque	NUMBER (15)

VILLE :

cville	ALPHA (3)
nom	ALPHA (20)

COMPTE :

num	NUMBER (12)
cdev	NUMBER (3)
cserv	FIELD (16)
renseignements	ALPHA (132)

BANQUE/PAYS :

renseignements	ALPHA (132)
ident. siège princ.	ALPHA (10)
raison sociale	ALPHA (60)

CORRESPONDANT :

identificateur	ALPHA (10)
raison sociale	ALPHA (15)
renseignements	FIELD (4)
cserv	FIELD (16)

ADRESSE TELEX :

identificateur	ALPHA (10)
cserv	FIELD (16)
échange de clé	FIELD

ADRESSE SWIFT :

identificateur	ALPHA (11)
authenticateur	FIELD
active	FIELD
cserv	FIELD (16)
date dern modif	NUMBER 98)

ANCIENNE ADRESSE SWIFT :

date de maj	NUMBER (8)
identificateur	ALPHA (10)

ANCIEN COMPTE :

date de maj	NUMBER (8)
numéro	NUMBER (12)
cdev	NUMBER (3)

ADRESSE POSTALE :
---

intitulé	ALPHA (30)
échange signature	FIELD
cserv	FIELD (16)

ADRESSE TELEGRAPHIQUE :
---

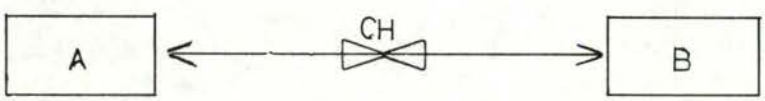
identificateur	ALPHA (20)
cserv	FIELD (16)

Annexe 4 : Transformations de structures

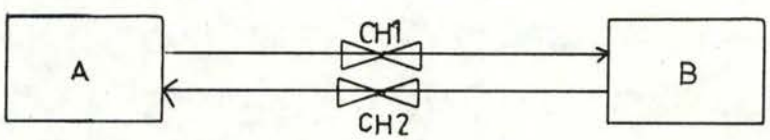
Nous allons ici, présenter la formalisation d'un certain nombre de transfromations que nous utiliserons tout au long de cette étude . Pour trouver une description complète de ces transformations , on se rapportera à [3].

1. Introduction / Suppression d'un type d'article

Soit CH une relation d'accès entre les types d'article A et B.



Cette relation peut s'exprimer par un chemin d'accès CH1, doté de son inverse CH2



Cette notion de chemin inverse peut se décrire ainsi :

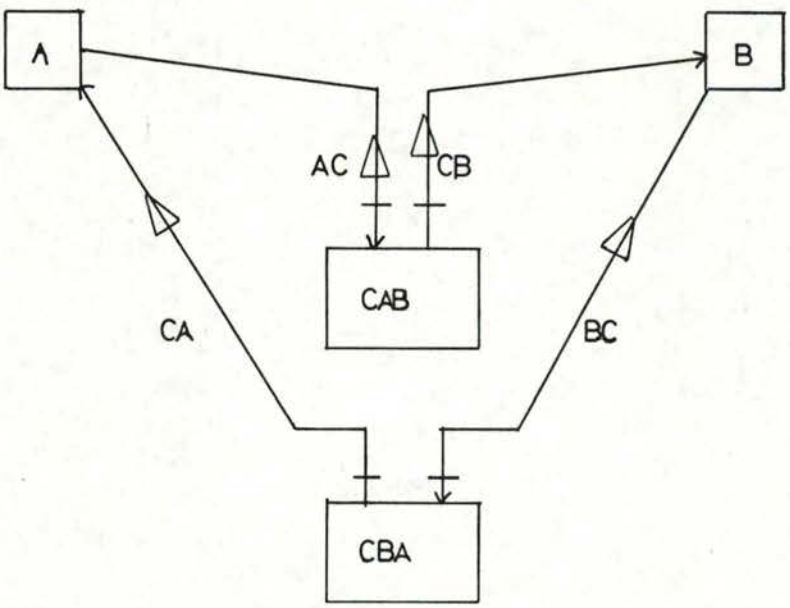
-tout article origine d'un chemin CH1 est cible d'un chemin CH2.
-tout article cible d'un chemin CH1 est origine d'un chemin CH2.
-tout article origine d'un chemin CH2 est cible d'un chemin CH1.
-tout article cible d'un chemin CH2 est origine d'un chemin CH1.

Les permutations sont celles de son inverse à la permutation près des cibles et des origines.

En particulier,

si un type est	alors son inverse est
M-N	N-M
1-N	N-1
N-1	1-N
1-1	1-1
fort pour l'origine	fort pour la cible
fort pour la cible	fort pour l'origine

On peut transformer la deuxième représentation par :



Cette transformation obéit aux règles énoncées ci-après

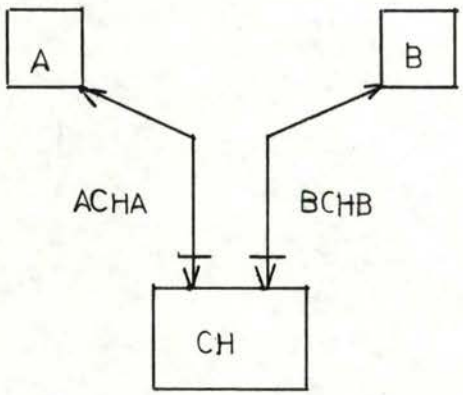
CH1 sera transformée comme ceci .

Pour tout b appartenant à A :
Pour chaque occurrence de chaque chemin d'origine a et de cible b, créer un article du type CAB et créer un chemin du type AC de a vers cet article et un chemin du type CB de cet article vers b.

CH2 sera transformée ainsi :

Pour tout b appartenant à B :
Pour chaque occurrence de chaque chemin d'origine b et de cible a, créer un article du type CBA et créer un chemin du type BC de b vers cet article et un chemin du type CA de cet article vers a.

On peut représenter la transformation ainsi :



En effet, de par les contraintes vues ci-dessus, les deux types d'articles CAB et CBA sont identiques et les types de chemin CA, BC sont respectivement inverses de AC, CB.

Nous venons, ici, de transformer un type de chemin M-N en la composition de deux types de chemin respectivement 1-N et N-1.

2.Définition de la composition de chemin.

Un type de chemin composé est un type de chemin fictif constitué par un enchaînement de types de chemin réels notés CH1, CH2,...Chn.

Le type de chemin est défini comme suit :

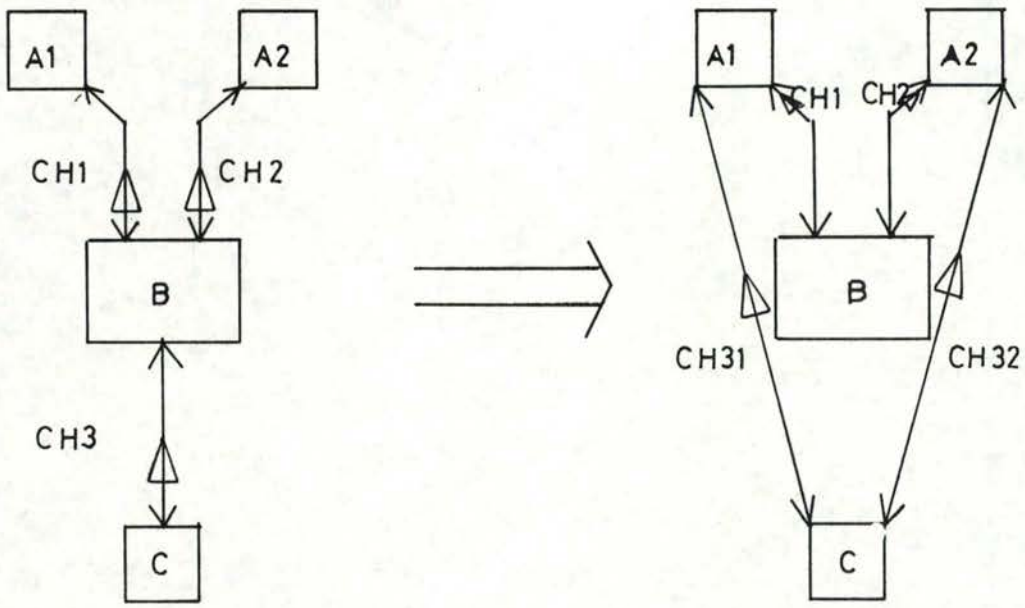
a) l'origine du type de chemin fictif est l'origine de CH1
b) la(es) cible(s) du type de chemin est(sont) la(es) cible(s) de CH2.
c) l'origine de Chi est la cible de CHi-1 et la cible de CHi est l'origine de CHi+1 pour i = 2,...,n-1;
d) Il n'est pas défini sur un chemin fictif d'autres opérations que l'accès aux cibles successives.

On réobtiendra la structure de départ en composant

ACHA o BCHB pour obtenir CH2
BCHB o ACHA " " CH1

en supprimant le type d'article CH.

3.Migration d'une relation



1 2

Nous ne donnerons plus ici, que les règles de passage.

1 ===> 2 CH31 = CH1 o CH3
CH32 = CH2 o CH3

en respectant la notion de chemin inverse.
Avec comme contrainte :
l'ensemble des articles du type B cibles des chemins du type CH3 doit être inclus dans l'ensemble des articles du type B origines des chemins du type CH1 et dans l'ensemble des articles du type B origines des chemins du type CH2.

1 <=== 2 CH3 = CH31 o CH1 ∩ CH31 o CH2

Avec comme contrainte :
l'ensemble des articles origines ou cibles des chemins du type CH31 doit être égal à l'ensemble des articles origines ou cibles des chemins du type CH32.

CH31 o CH32 $\subseteq$ CH1 o CH2
CH32 o CH31 $\subseteq$ CH2 o CH1

Annexe 5 : Estimation du volume de la base.

Type d'article PAYS.

200 articles * 71 car = 2366 mots (1 mot = 6 car.)

renseignements-p (lien)

200 mots

access-ville (subset)

200 mots (adresses des tables)
1000 mots (adresses des articles villes)

access-pays-INS (set)

200 mots (adresses des pays)
100 mots (valeurs de clé)

access-pays-IBIC

200 mots (adresses des pays)
100 mots (valeurs de clé)

Type d'article VILLE.

1000 * 23 car. = 3833 mots

access-corresp

1000 mots (adresses des tables)
6000 mots (adresses des correspondants)

access-ville
---

500 mots (1000 * 3 car.)
1000 mots (adresses des villes)

Type d'article PAYS-R.

2200 mots (100 * 22 mots)

Type d'article BANQUE/PAYS.

36500 mots (3000 * 73 car.)

renseignements-b-p

1500 mots (adresses des renseignements)

access-correspondant

3000 mots (adresses des tables)
6000 mots (adresses des correspondants)

Type d'article BANQUE/PAYS-R.

3300 mots (1500 * 22 mots)

Type d'article BANQUE.

27333 mots (2000 * 82 car.)
access-banque-pays

2000 mots (adresses des tables) 3000 mots (adresses des banques-pays)

Type d'article CORRESPONDANT.

29000 mots (6000 * 29 car.)

access-banque-pays

6000 mots (adresses des correspondants)

access-adresse-swift-utl
---

on ne connait pas précisément le nombre d'articles cibles de ce type de chemin, fixons le à 10000, ce qui semble une limite raisonnable.
6000 mots (adresses des tables)
10000 mots (adresses des articles cibles)
3333 mots (10000 * 2 car. (cserv))

access-adresse-swift-poss

6000 mots (adresses des tables)
3800 mots (pointeurs vers les adresses swift)

access-compte-prop

6000 mots (adresses des tables)
5200 mots (adresses des comptes)

access-compte-utlc
---

on ne connait pas précisément le nombre d'article cible de ce chemin, fixons le à 10000 ce qui semble un nombre raisonnable.
6000 mots (adresses des tables) 10000 mots (adresses des comptes)

Type d'article ADRESSE TELEX
---

14 900 mots (7000 * 12.5 car)

access-adresse-telex

7000 mots (pointeurs vers les adresses télex)

2333 mots (7000 occurrences du code service (2 car.))

Type d'article ADRESSE POSTALE

48 700 mots (9000 * 32.5 car) access-adresse-postale

9000 mots (pointeurs vers les adresses postales)
3000 mots (9000 occurrences du code service (2 car.))

Type d'article ADRESSE TELEGRAPHIQUE

25 666 mots (7000 * 22 car.)

access-adresse-télégraphique

7000 mots (pointeurs vers les adresses télégraphiques)
2333 mots (7000 occurrences du code service (2 car.)

Type d'article COMPTE

14 733 mots (5200 * 17 car.)

renseignements-c (link)

5200 mots (adresses des renseignements)

access-correspondant-prop (link)

5200 mots (adresses des correspondants propriétaires)

access-correspondant-dep

5200 mots (adresses des correspondants dépositaires)

access-ancien-compte

5200 mots (adresses des tables des articles cibles)

On ne dispose pas de renseignements précis sur la taille des tables.

Type d'article COMPTE-R
---

57 200 mots (2600 * 132 car.)

Type d'article ADRESSE SWIFT
---

14 000 mots (3800 * 22 car.)

access-ancienne-adresse

3800 mots (adresses des tables des articles cibles)

On ne dispose pas de renseignements précis sur la taille des tables.

access-correspondant

3800 mots (adresses des correspondants propriétaires)

Types d'article ANCIEN COMPTE, ANCIENNE ADRESSE SWIFT

On ne dispose pas de renseignements précis sur ces types d'article.

CONCLUSION :

Le volume total de la base est de 3 000 000 de caractères, ce qui est relativement peu. Nous pouvons donc en déduire qu' une solution privilégiant le temps d'accès est envisageable.

BUMP
0 0 3 2 2 0 8 1 0
*FM B16/1981/03

Mémoire
Collin.
1981.
Compliments.

A. INTRODUCTION

Dans cet article, nous allons comparer la proposition de base de données proposées dans "Conception de la base de données d'un système d'information d'un organisme bancaire" et que nous rappelons en annexe avec celle d'veloppée par la banque Bruxelles-Lambert. Avant cette comparaison, nous fixerons les paramètres physiques de la base. Pour cette analyse, nous ne reprendrons pas, ici, toutes les caractéristiques des types d'articles et de chemins. Nous nous reporterons pour cela à la description présentée dans le mémoire sus-mentionné.

B. FIXATION DES PARAMETRES PHYSIQUES

Dans cette partie, nous allons déterminer le nombre d'articles par bloc pour les types d'article et le nombre d'entrées par table pour les sets et subsets. Nous trouverons, à la fin de cet articles quelques précisions sur l'organisation des types d'article et des types de chemin. Nous procèderons, en reprenant chaque type de chemin et chaque type d'article et en précisant pour chacun de ceux-ci la valeur retenue des paramètres.

REMARQUES PRELIMINAIRES :

Dans la suite, nous allons parler de segment. Il faut savoir que celui-ci est l'unité de transfert entre mémoire principale et mémoire secondaire. Sa taille set de 30 mots ou 180 caractères. Il est difficile de déterminer le meilleur compromis : transférer un grand nombre de segments en une fois ou effectuer plusieurs transferts d'un petit nombre de segments. on peut dire, cependant que dans le cas d'un accès séquentiel aux articles d'un type, on a avantage à transférer le plus grand nombre possible d'articles. Dans le cas d'accès aléatoire, on aura souvent intérêt à ne pas avoir des blocs contenant trop d'articles, du moins si la probabilité de se retrouver dans ce bloc pour un accès ultérieur est faible.

LBS 6544799
296768

access-ville (SUBSET)

Pour les raisons définies par ailleurs (améliorer le temps d'accès), nous avons décidé de retenir un lien et non plus un subset pour implémenter ce type de chemin. En effet, on économise au minimun une lecture physique par article origine d'un chemin, celle de la table. De plus, en raison de la consommation d'un segment minimum par chemin, la place économisée par un subset est beaucoup moins importante. Nous avons fixé le nombre maximun d'articles cibles de ce type de chemin à 20.

PAYS (DATA SET)

Nous avons 13 mots pour les items , 20 mots pour le lien access-ville, 1 mot pour le lien renseignements-p. Ce qui nous donne 34 mots par article. Le type d'accès étant essentiellement séquentiel, nous prendrons un facteur de blocage relativement grand, soit 14 articles par bloc. Nous occuperons, pour ces 14 articles 476 mots + 1 mot de controle par bloc : soit 477 mots ou 16 segments (480 mots).

access-pays-INS (SET)

Par entr'e, nous avons 1 cpaysINS (3 car.), 1 mot d'adresse, 1 mot pour résoudre les duplicates, soit 3 mots par entrée. Nous avons, en plus, deux mots de contrôle par table.

Nous limiterons le nombre de niveaux à deux en utilisant la formule énoncée en annexe.

nb N > (P ** (1/n) / L

N : nbre entrée par table
P : nbre d'article du type sur lequel est défini le set, subset
n : nbre de niveaux de table.
L : taux de chargement des tables

N > (200 ** (1/2) / 0.8
donne un N de 18.

Le type d'accès étant essentiellement aléatoire on n'a pas avantage a fixer le nombre d'entrée par table trop haut, nous retiendrons 19. Nous aurons donc 19 * 3 mots + 2 mots de contrôle, soit 59 mots ou (2 segments - 60 mots).

access-pays-IBLC.

Par entrée : cpaysIBLC (3 car.), 1 mot d'adresse, soit 2 mots. Nous tiendrons le même raisonnement pour le nombre de niveaux que celui tenu pour access-pays-ins. Le type d'accès étant toujours aléatoire nous fixerons, ici, le nombre d'entrée par table à 29 et ce afin de ne pas trop gaspiller de place (29 * 2 mots + 2 mots = 60 mots)

VILLE (data set)

Nous avons 5 mots pour les items, 1 mot pour le subset access-correspondant ,soit 6 mots.

Le type d'accès étant des deux natures (séquentiel et aléatoire), nous retiendrons un facteur de blocage moyen, soit 34 articles par bloc. Un bloc occupera donc 34 * 6 mots, soit 204 mots + 1 mot de contrôle, soit 7 segments.

access-corresp (SUBSET)

Nous fixerons la taille des tables au nombre minimum, soit un segment En effet, nous ne disposons pas d'informations suffisantes sur ce type de chemin et 1 segment permettra d'accéder à 28 correspondants par ville.

access-ville (SET)

Par entrée : cville (3 car.), 1 mot d'adresse, soit deux mots. En applicant la formule pour limiter a deux niveaux, on obtient qu'il faut plus de 39 entrées par table. Le type d'accès étant essentiellement aléatoire, nous fixerons ce nombre à 44, soit 44 * 2 : 88 mots + 2 mots de controle = 90 mots, soit 2 segments.

PAYS-R (data set)

Nous avons 22 mots pour les items, 1 mot de count, soit 23 mots. Le type d'accès etant essentiellement séquentiel, nous essayerons d'avoir un facteur de blocage assez important. Nous retiendrons 38 articles par bloc, soit 836 mots + 1 mot de controle, soit 837 mots ou 28 segments (840 mots).

BANQUE-PAYS-R (data set)

Nous tiendrons le même raisonnement que pour PAYS-R.

COMPTE-R (data set)

Nous tiendrons le même raisonnement que pour PAYS-R.

access-correspondant (SUBSET)

Pour les mêmes raisons que dans le cas d'access-ville, on remplacera le subset par un lien. On fixera le nombre maximun d'articles cibles des chemins de ce type à 10.

BANQUE-PAYS (data set)

Nous avons 13 mots pour les items, 10 mots pour le lien, soit 23 mots par article. Les deux type d'accès sont requis. nous fixerons donc le facteur de blocage à 22 articles, soit 506 mots + 1 mot de controle, soit 17 segments.

BANQUE (data set)

Pour les items : 16 mots, 1 mot pour le subset, soit 17 mots. Le type d' accès étant essentiellement séquentiel, nous fixerons un facteur de blocage assez élevé : 26 articles par bloc, soit 442 mots + 1 de controle, soit 15 segments.

access-banque-pays (SUBSET)

Ne disposant pas suffisamment d'informations précises sur ce type de chemin, nous fixerons la taille des tables à 1 segment.

CORRESPONDANT (data set)

Pour les items : 6 mots, 5 mots pour le lien et les subsets, 3 mots pour les pointeurs vers les articles cibles des chemins définis par les types d'articles imbriqués, soit 14 mots. Les deux types d'accès sont nécessaires , nous fixerons donc le facteur de blocage à 21 articles, soit 294 mots + 1 mot de controle, soit 10 segments.

access-adresse-swift-utl (SUBSET)

Par enterée : cserv (field (16)), 1 mot d'adresse, soit 2 mots Nous fixerons la taille des tables à 1 segment car nous n'aurons sans doute jamais plus de 14 adresses swift utilisables par un même correspondant.

access-adresse-swift-poss (SUBSET)

Pour les mêmes raisons que dans le cas d'adresse-swift-utl,nous retiendrons 1 segment par table.

access-compte-prop (SUBSET)

Pour les mêmes raisons que dans le cas d'adresse-swift-utl, nous retiendrons 1 segment par table.

access-compte-utlc (SUBSET)

Pour les mêmes raisons que dans le cas d'adresse-swift-utl, nous retiendrons 1 segment par table.

ADRESSE-TELEX (data set)

Nous avons 3 mots pour les items.Nous avons, ici, affaire à un type d'article imbriqué, nous emploierons le type d'organisation STANDARD qui ne gaspillera pas trop de place car il peut y avoir dans un bloc des articles appartenant à des chemins différents. Le type d'accès étant essentiellement aléatoire, nous ne retiendrons pas un facteur de blocage trop élevé, soit 9 articles ou 1 segment.

access-adresse-telex (SET)

Nous retiendrons, ici, l'organisation ORDERED LIST et nous fixerons la taille des tables à 1 segment, dans une table ne pouvant se trouver que les articles appartenant au même chemin.

ADRESSE-POSTALE (data set)

Nous avons, pour les items, 6 mots. Le type d'accès étant essentiellement aléatoire, nous fixerons la taille des blocs à 9 articles, soit 2 segments. Pour les mêmes raisons que dans le cas d'adresse telex nous retiendrons l'oorganisation STANDARD.

access-adresse-postale (SET)

Nous retiendrons les conclusions d'access-adresse-telex.

ADRESSE-TELEGRAPHIQUE (data set)

Nous avons, pour les items, 4 mots.Nous retiendrons le même type d'organisation qu'ADRESSE-TELEX et nous fixerons le facteur de blocage à 7 articles, soit 1 segment.

access-adresse-telegraphique (SET)

Nous retiendrons les mêmes conclusions qu'access-adresse-telex.

COMPTE (data set)

Nous avons, pour les items, 3 mots et 4 mots pour les liens et subset. Le type d'accès étant essentiellement aléatoire, nous fixerons le facteur de blocage à 4, soit 1 segment.

access-compte-ancien-compte (SUBSET)

Ne disposant pas d'informations précises sur ce type de chemin, nous fixerons la taille des tables à 1 segment.

access-compte (SET)

Par entrée nous avons 3 mots pour le cdev et le numéro, 1 mot d'adresse, soit 4 mots.Nous avons retenu l'organisation INDEXED-SEQUENTIAL et nous limiterons le nombre de niveaux de tables à 2 en appliquant la formule déja énoncée plus haut.

N > 5200 ** (1/2) / 0.8
N doit être supérieur à 91

Nous retiendrons un N de 119 qui limite la perte de place (16 segments).

ANCIENNE-ADRESSE-SWIFT (data set)

Nous avons 4 mots pour les items + 1 mot d'adresse pour le subset. Ne disposant que de peu d'informations, nous fixerons la taille des blocs à 1 segment, soit 5 articles.

access-ancienne-adresse-swift (SET)

Par entrée, on aura 2 mots, pour l' identificateur, 1 mot d'adresse, soit 3 mots. Ne disposant pas d'informations précises sur le nombre d'articles de ce type, il nous est difficile de limiter le nombre de niveaux à 2. Nous retiendrons une taille de table de 3 segments.

ANCIEN-COMPTE (data set)

Nous avons pour les items 6 mots, 1 mot pour le type de chemin vers COMPTE , soit 7 mots. Ne disposant pas d'informations précises sur ce type d'article , nous fixerons le nombre d'articles par bloc à 8, soit 2 segments.

access-ancien-compte-compte (SUBSET)

Ne disposant pas d'informations précises sur ce type de chemin, nous fixerons la taille des tables à 1 segment.

access-ancien-compte (SET)

Par entrée, nous avons 3 mots pour le cdev et le numéro et 1 mot d'adresse, soit 4 mots. Nous avons retenu l'organisation INDEXED-SEQUENTIAL et comme nous ne disposons pas d'informations suffisantes pour limiter le nombre de niveaux d'index à 2, nous fixerons le nombre d'entrée par table à 21, soit 3 segments.

ADRESSE-SWIFT (data set)

Nous avons pour les items 4 mots, 2 mots pour les lien et subset, soit 6 mots. Le type d'accès étant essentiellement aléatoire, nous fixerons le facteur de blocage à 19 articles, soit 114 mots + 2 mots de controle, soit 4 segments.

access-adresse-swift (SET)

Par entrée, nous avons 2 mots pour l'identificateur, 1 mot d'adresse, soit 3 mots. Nous limiterons le nombre de niveaux de table en fixant le nombre d'entrée par table comme précédemment.

N > 3800 ** (1/2) / 0.8
N doit donc tre supérieur à 78. Nous retiendrons donc N = 99, 10 segments.

access-correspondant (SET)

PaPar entrée, nous avons 2 mots pour l'identificateur, 1 mot d'adresse, soit 3 mots. Nous limiterons le nombre de niveaux de tables en fixant N

N > 6000 ** (1/2) / 0.8
N doit donc être supérieur à 97 e Nous retiendrons donc N = 128, soit 13 segments.

access-ancienne-adresse-swift-adresse-swift (SUBSET)

Ne disposant pas d'informations précises sur ce type de chemin, nous fixerons la taille des tables à 1 segment.

access-adresse-swift-ancienne-adresse-awift (SUBSET)

Ne disposant pas d'informations précises sur ce type de chemin, nous fixerons la taille des tables à 1 segment.

C. ESTIMATION DU TEMPS D'ACCES DES APPLICATIONS 1 ET 7

Nous allons, ici, estimer le temps d'accès des deux applications les plus utilisées, Ne disposant pas de renseignements précis sur les disques du système (expl : le nombre de segments par piste) , nous considérerons un temps moyen de transfert quelque soit le nombre de segments (nous n'avons pas proposé des blocs supérieur à 700 mots) égal à 50 milisecondes.

APPLICATION 1 :

Accès compte (numéro,cdev) de type aléatoire. 2 accès sont nécessaires; 1 à la table du set pour obtenir l'adresse physique de l'article compte et 1 pour obtenir cet article.

Accès correspondant (prop). 1 accès car dans l'article compte, nous disposons de l'adresse physique de l'article "correspondant".

Accès adresses. Dans 70 % des cas, 2 accès suffiront : 1 accès à la table du subset qui définit le type de chemin "propriétaire d'adresse swift" et 1 accès à l'article en question. Dans 20 % des cas, nous aurons 3 accès : le premier cité + 1 accès à la table du subset qui définit le type de chemin "utilisateur de l'adresse swift" et 1 à l'article en question. Dans quelques cas, nous aurons 4 accès physiques : les premiers cités des deux premiers cas + 1 accès à la table du subset qui définit le type de chemin "propriétaire d'adresse telex" et 1 accès à cet article.

Nous considérerons donc un nombre moyen de 6 accès physiques, ce qui donne pour 1600 exécutions par jour 1600 * 300 ms, soit 480000 ms, soit 8 minutes d'opérations d'entrées-sorties.

Si l'on répète le même raisonnement avec l'application 7, on obtient en moyenne 7 accès physiques, soit un temps d'opérations d'entrées-sorties de 350 ms * 1600, soit environ 10 minutes.

Le temps d'accès semble, donc à première vue raisonnable. Il faudrait maintenant examiner les heures auxquelles sont exécutés ces applications et la charge du système à ces heures pour voir si le temps de réponse peut être considéré comme acceptable. Nous n'avons pu encore réaliser ce travail.

D. COMPARAISON ENTRE LES SOLUTIONS

Pour comparer les deux solutions, il nous faut nous définir une grille de critères de comparaison. Nous pourrions reprendre la grille des critères qui ont servi à l'élaboration de notre base. Cette grille n'étant pas identique à celle sous-tendant la solution de la banque , nous nous contenterons donc, de comparer ici, les grandes options de conception. Il est d'ailleurs très difficile de comparer les deux propositions. En effet, celle retenue par la banque tient compte de l'environnement existant et de la présence d'autres systèmes d'information. Nous n'avons pas tenu compte de ceci dans notre étude. Pour mettre en évidence ce fait, nous prendrons l'exemple des adresses postales et télex qui sont présentes dans notre base mais qui dans celle de la BBL sont remplacées par des numéros qui sont des clés permettant d'accéder à ces adresses dans les fichiers qui les contiennent (TELEX, CLIENT ...).

Une grande différence entre les deux propositions est la prise en compte de la notion de correspondants et d'adresses swift. Dans la proposition de la banque , cette notion a été , nous semble-t-il, noyée. Ces deux types d'entités sont en effet, fondus en un seul avec pour conséquence que l'on a dû créer des adresses swift fictives pour désigner des correspondants qui n'en possédaient pas et que plusieurs occurrences de ce type mixte désignent le même correspondant. Ceci nous semble une erreur, car s'éloigner à ce point du schéma conceptuel ne peut amener que la confusion.

Une autre grande différence des deux propositions est l'emploi d'une clé (cserv) dans notre solution pour l'obtention des différentes adresses ou comptes d'un correspondant. Lors de l'analyse conceptuelle, on avait mis en évidence que , pour un service déterminé, une seule adresse de chaque type et un seul compte pouvait exister. La solution de la banque nécessite la lecture des différentes adresses ou des différents comptes d'un correspondant afin de déterminer celui ou celle utilisable par un service déterminé.

Une dernière grande différence réside dans le processus de traduction des types de chemin. Les services de la BBL ont traduit la plupart de ceux-ci par des migrations d'items ou par la création de types d'article. Nous avons proposé ce type d'implémentation à plusieurs reprises; nous l'avons peu retenu , eu égard à notre grille de critères. Nous pouvons difficilement juger cette différence eu

égard à l'expérience approfondie des responsables de la banque du système de gestion de base de données. Nous sommes quant à nous partis du postulat suivant : les mécanismes de DMS offrent les services et les sécurités décrits dans les manuels de référence.

Nous n'avons certainement pas tiré toutes les conclusions de ce travail, mais nous avons effectué ce que nous pouvions faire dans le temps que nous disposions.

PRECISION SUR L'ORGANISATION PHYSIQUE DES TYPES D'ARTICLES , DES SETS ET SUBSETS

1) Type d'article :

a) Organisation ORDERED

Ce type d'organisation est surtout adapté aux types d'article imbriqués . Deux cas se présentent :

1) non-emploi de sous-blocs
2) emploi de sous-blocs

1) Dans un bloc , on ne trouvera que les articles cibles du même chemin. Ceux-ci sont ordonnés sur les valeurs de la clé définie par l'ACCESS (obligatoire).

2) Un bloc est organisé en sous-blocs. Dans un sous-bloc, tous les articles sont cibles du même chemin. Dans un bloc, on trouvera des articles cibles de chemins différents.

On jouera sur le paramètre taille des blocs / sous-blocs suivant le type d'accès , afin d'améliorer la combinaison espace mémoire / temps d'accès et de traitement. On peut dire que si l'on accède aux articles d'un type séquentiellement, on a avantage à augmenter le nombre d'articles par bloc et à avoir des sous-blocs.

Enfin, on ne peut définir de set ou de subset sur les types d'article organisés de cette façon. Deux mots de contrôle sont adjoints à chaque bloc.

b) Organisation UNORDERED

Ce type d'organisation convient surtout aux types d'articles imbriqués. Dans un bloc, on ne peut trouver que des articles cibles du même chemin. On ne retrouve pas ici la notion de sous-blocs. Les espaces libres contigus dans un bloc sont fusionnés et sont maintenus par une liste chaînée . Deux mots par bloc sont réservés au système . Les articles ne sont pas maintenus par ordre de valeurs de clés.

c) Organisation STANDARD

Organisation par défaut des types d'article. Les articles ne sont pas maintenus en ordre logique. Les blocs de données contiennent des articles et des espaces libres (articles supprimés). Ces espaces libres sont retrouvés par l'intermédiaire de tables figurant dans des blocs réservés dont une partie réside en mémoire centrale dès l'ouverture de la base. Quand un article est supprimé , son adresse est placée dans la table qui est organisée comme une pile. Ce type d'organisation utilise efficacement l'espace disque. En effet, un nouveau bloc n'est alloué qu'à l'épuisement des précédents. Si le type d'article est imbriqué , un set ou un subset doit être déclaré sur celui-ci car les articles cibles d'un même chemin peuvent appartenir à des blocs différents , un même bloc pouvant contenir des articles cibles de chemins différents.

2) Type de chemin (SET - SUBSET)

a) Organisation INDEXED SEQUENTIAL

Collection de tables organisée hiérarchiquement . Au niveau le plus bas, on trouve des tables fines. Dans ces tables, il existe une entrée pour chaque article du type d'article sur lequel le set ou le subset est défini. Une entrée est composée de la valeur de clé et de l'adresse d'un article. Dans chaque table, les entrées sont conservées par valeur croissante de clé. Les tables fines sont organisées par un autre niveau de tables. Si plus d'une table est nécessaire pour adresser l'ensemble des tables fines, un niveau supplémentaire est créé qui adressera le deuxième niveau de tables et ainsi de suite jusqu'à ce qu'une table suffise au dernier niveau. Nous essayerons ici, de ne pas dépasser 2 niveaux de table, la table du dernier niveau résidant en mémoire centrale, il faudra deux accès physiques pour obtenir un article.

Le nombre de niveaux peut être contrôlé par le nombre d'entrées par table. Posons le nombre d'entrées par table égale à N, le nombre de niveaux étant représenté par n et le nombre d'articles du type sur lequel est défini le set ou subset par P, L étant le taux de chargement des tables.

Le nombre d'articles adressables par 1 niveau de table = N
Le nombre d'articles adressables par 2 niveaux de table = N*N = N**2
Le nombre d'articles adressables par n niveaux de table = N**n

Si l'on tient compte du taux de chargement , nous avons N**n * L
Nous avons donc :

P = N**n * L
N**n = P /L
N = P** (1/n) / L

Il faut donc que N > P** (1/n) /L pour limiter le nombre de niveaux à n.

N.B. ** symbolise l'exposition
N**2 = N au carré.

b) Organisation ORDERED LIST

Les entrées sont conservées par ordre de valeurs de clé. Les entrées valides sont toujours contiguës dans les tables. Les tables sont chaînées sous la forme d'une liste circulaire à deux sens. Dans ces chaînes, les tables sont maintenues en séquence. Une table ne peut contenir que les entrées des articles cibles d'un chemin et il y a une chaîne par article origine d'un chemin. Comme les entrées sont maintenues en séquence, une entrée doit se trouver à une position particulière. Si une table est pleine , il faut donc l'éclater en deux autres. Le paramètre LOADFACTOR fixe le nombre d'entrées transférées dans le second bloc.

c) Organisation UNORDERED LIST

Organisation conçue pour les types d'article imbriqués. les articles sont maintenus par ordre d'adresse physique. Pour le reste, les caractéristiques sont identiques à l'organisation ORDERED LIST.

Schéma DDL de la base "correspondants".

```
PAYS DATA SET


(clangue              NUMBER (2)  OCCURS  3 TIMES;
cdev                  NUMBER (3);
nom-français          ALPHA (20);
nom-néerlandais       ALPHA (20);
cpays-RTT             ALPHA (3);
cpays-INS             NUMBER (3);
cpays-ISO             ALPHA (2);
cpays-IBLC            NUMBER (3)  REQUIRED;
limite-risque         NUMBER (15);
renseignements-p      REFERENCE TO   PAYS-R  COUNTED;
access-ville          REFERENCE TO VILLE COUNTED OCCURS 20 TIMES;
);

access-pays-INS SET OF PAYS
KEY IS cpays-INS
DUPLICATES;

access-pays-IBLC SET OF PAYS
KEY IS  cpays-IBLC
NO DUPLICATES;


VILLE DATA SET


(count-ville          COUNT (100) REQUIRED;
cville               ALPHA (3)  REQUIRED;
nom                   ALPHA (20);
access-corresp SUBSET OF CORRESPONDANT;
);


Access-ville SET OF VILLE
KEY IS cville
NO DUPLICATES;
```

```
PAYS-R DATA SET


(renseignements      ALPHA (132)  REQUIRED;
count-pays-r         COUNT (100)   REQUIRED;
);


BANQUE-PAYS-R DATA SET


(renseignements      ALPHA (132)  REQUIRED;
count-banque-pays-r COUNT (100)  REQUIRED;
);


BANQUE-PAYS DATA SET


(identif.-siège-principal     ALPHA (10);
raison-sociale                ALPHA (60);
renseignements-b-p  REFERENCE TO BANQUE-PAYS-R COUNTED;
cpays-IBLC          NUMBER (3)  REQUIRED;
access-correspondant  REFERENCE TO CORRESPONDANT OCCURS 10 TIMES;
);


BANQUE DATA SET


(cbanque             ALPHA (4)  REQUIRED;
raison-sociale       ALPHA (60);
cpays-origine        NUMBER (3);
limite-risque        NUMBER (15);
access-banque-pays   SUBSET OF BANQUE-PAYS;
);


CORRESPONDANT DATA SET


(identificateur      ALPHA (10)  REQUIRED;
count-correspondant COUNT (100) REQUIRED;
raison-sociale       ALPHA (15);
renseignements       FIELD (4);
cville               ALPHA (3)  REQUIRED;
access-correspondant-banque-pays  REFERENCE TO BANQUE-PAYS COUNTED;
```

```
access-adresse-swift-utl  SUBSET OF ADRESSE-SWIFT KEY IS cserv;
access-adresse-swift-poss SUBSET OF ADRESSE-SWIFT KEY IS cserv;
access-compte-prop  SUBSET OF COMPTE KEY IS cserv, cdev;
access-compte-utlc  SUBSET OF COMPTE KEY IS cserv, cdev;

access-correspondant SET OF CORRESPONDANT
KEY IS identificateur
NO DUPLICATES;


ADRESSE-TELEX DATA SET


(identificateur     ALPHA (10) REQUIRED;
cserv               FIELD (16) REQUIRED;
échange-de-clé      FIELD;
);

access-adresse-télex  SET OF ADRESSE-TELEX KEY IS cserv;


ADRESSE-POSTALE DATA SET


(intitule           ALPHA (30)  REQUIRED;
échange-signature   FIELD;
cserv               FIELD (16)  REQUIRED;
);

access-adresse-postale  SET OF ADRESSE POSTALE KEY IS cserv;


ADRESSE TELEGRAPHIQUE


(identificateur     ALPHA (20)  REQUIRED;
cserv               FIELD (16)  REQUIRED;
);

access-adresse-télégraphique  SET  OF  ADRESSE-TELEGRAPHIQUE  KEY   IS
cserv;

);


COMPTE DATA SET


(numéro             NUMBER (12)  REQUIRED;
cdev                NUMBER (3)   REQUIRED;
```

```
cserv                FIELD (16)   REQUIRED;
renseignements-c  REFERENCE TO COMPTE-R COUNTED;
access-correspondant-prop  REFERENCE TO CORRESPONDANT COUNTED;
access-correspondant-dep  REFERENCE TO CORRESPONDANT COUNTED;
access-compte-ancien-compte  SUBSET OF COMPTE;
);


access-compte SET OF COMPTE
KEY IS cdev, numero
NO DUPLICATES;


ANCIENNE-ADRESSE-SWIFT


(identificateur      ALPHA (10)  REQUIRED;
date-de-maj          NUMBER (8)  REQUIRED;
access-ancienne-adresse-swift-adresse-swift  SUBSET OF ADRESSE-SWIFT;
);

access-ancienne-adresse-swift SET OF ANCIENNE-ADRESSE-SWIFT
KEY is identificateur
NO DUPLICATES;


ANCIEN-COMPTE


(numéro              NUMBER (12)  REQUIRED;
cdev                 NUMBER (3)   REQUIRED;
date-de-maj          NUMBER (8)   REQUIRED;
access-ancien-compte-compte  SUBSET OF COMPTE;
);

access-ancien-compte SET OF ANCIEN COMPTE
KEY IS numéro , cedv
NO DUPLICATES;


COMPTE-R  DATA SET


(renseignements      ALPHA (132)  REQUIRED;
count-compte-r          COUNT (100)   REQUIRED;
);
```

```
ADRESSE-SWIFT DATA SET


(identificateur      ALPHA (11)  REQUIRED
authenticateur       FIELD;
cactive              FIELD;
cserv                FIELD (16)  REQUIRED;
date-dern-modif      NUMBER (8);
access-adresse-swift-ancienne-adresse-swift    SUBSET   OF    ANCIENNE
ADRESSE SWIFT;
access-correspondant-prop  REFERENCE TO CORRESPONDANT COUNTED;
);


access-adresse-swift SET OF ADRESSE-SWIFT
KEY IS identificateur
NO DUPLICATES;
```

BUMP
0 0 6 5 4 4 7 9 9
*FM B16/1981/03S