# Structuration conceptuelle et physique de l'espace des objets dans un Système de Gestion des Données et des Connaissances

Ana Simonet, Michel Simonet

TIMC-IMAG, Université Joseph Fourier
Grenoble, France
{Ana.Simonet, Michel.Simonet}@imag.fr

**Abstract.** Knowledge Base Management Systems are not designed to manage large amounts of data in an efficient manner, contrary to DBMS of which it is the primary purpose. We present an approach that unifies KBMS and DBMS, by providing an efficient management of wide volumes of (shared) data and reasoning capabilities (instance classification). We propose a conceptual structuring of the object space that relies on logical properties used in the definition of classes, namely the Classification Space, which supports instance classification, integrity checking and object indexing through an appropriate physical representation. This representation also supports semantic query optimization and integration of heterogeneous data.

**Keywords:** Database, knowledge base, indexing, semantic query optimization, instance classification

## 1 Introduction

Les fonctions classiques d'un SGBD (Système de Gestion de Bases de Données) restent d'actualité, même si l'avènement des entrepôts de données a initié un nouveau paradigme, les bases de données OLAP (On-Line Analytical Processing), où l'accès rapide à de grandes masses de données devient un enjeu majeur. C'est dans la perspective des bases de données classiques (OLTP) qu'a été conçu le modèle de données dont il est question ici. Sa conception a été guidée par le souci de permettre le partage des données par diverses catégories d'utilisateurs, chacune les percevant selon ses points de vue. Les vues des bases de données relationnelles ont en quelque sorte « élargi » l'intention initiale des auteurs du modèle relationnel, pour qui les vues étaient une « fenêtre » sur les données [9]. Or les vues dans les systèmes actuels sont des requêtes qui peuvent retourner toute construction relationnelle qui peut être exprimée au moyen d'une requête SQL. Depuis, certains auteurs ont distingué les vues « object-preserving », qui ne peuvent retourner que des objets (ou des parties d'objets) existants [17] et les vues « object-constructing » qui peuvent créer de nouveaux objets [5]. Par ailleurs, du point de vue de la représentation physique des données, deux catégories de vues sont possibles : les vue virtuelles, qui sont les plus

habituelles, et les vues persistantes dont le développement se fait surtout autour des bases de données OLAP.

Le terme « base de connaissances », qui à l'origine, dans la décennie 70, désignait l'ensemble des règles d'un système expert (à base de règles), est aujourd'hui utilisé pour désigner une large gamme de représentations informatiques dont le seul dénominateur commun est de faire appel à du raisonnement logique. Ce même terme, dans la communauté française de l'IA des années 80, désignait principalement un système de type « frames » [13] dont l'objectif était de permettre le classement d'objet (ou classement d'instance), d'ailleurs improprement traduit par cette communauté par « classification[1] ». On peut remarquer que l'objectif d'un système expert à base de règles était de déterminer des conclusions en déroulant les règles, à partir de « faits » initiaux. Il s'agit là en fait d'une opération de classement, les faits initiaux représentant l'objet à classer et les conclusions les classes dont cet objet satisfait les propriétés.

Les communautés des bases de données et des bases de connaissances sont demeurées relativement séparées. Il y a bien eu quelques tentatives d'utilisation des systèmes à base de connaissances pour effectuer certains travaux sur les données des bases de données, par exemple pour le data mining ou l'intégration de données, mais les deux systèmes restaient distincts et il était nécessaire de convertir les données d'une représentation à l'autre, ce qui les limitait au rôle de prototype de recherche [6][7]. Aujourd'hui, les ontologies, qui sont les nouveaux représentants des systèmes à base de connaissances, sont abondamment présentes dans de nombreux travaux concernant les bases de données, mais il s'agit toujours de deux systèmes disjoints – le SGBD et le système de gestion d'ontologies – qui cohabitent et communiquent.

Or, il nous semble que le traitement des très grandes quantités de données et de connaissances disponibles dans des entreprises ou sur le web, nécessite des systèmes mixtes de type SGBD-BC, c'est-à-dire des systèmes intégrant les fonctionnalités des SGBD – notamment sa capacité de stockage de très grandes masses de données, de gestion de confidentialité et de cohérence –, ainsi que les fonctionnalités des SGBC, notamment le classement et d'autres formes de raisonnement. Le système Osiris est un SGBD-BC qui implémente le modèle des P-types.

Le modèle des P-types (P signifie *Partagé*) a été conçu avec une vision très orientée vers l'expression directe des besoins des utilisateurs, à travers un usage intensif des vues, qui représentent le point de vue d'un groupe homogène d'utilisateurs. Dans cette perspective, les « vues » constituent la définition même d'un P-type, et c'est le système – et non pas l'analyste comme dans l'approche classique de conception – qui détermine les tables appropriées[2] pour la gestion des données. Ainsi, un P-type est défini par la collection des points de vue qu'ont les utilisateurs sur les objets de ce type (cf §2). Les vues sont définies dans une hiérarchie de spécialisation

[1] Les termes *classement* et *classification* existaient déjà en analyse des données. Dans le domaine des bases de connaissances, la *classification de concepts (ou de classes)* désigne le calcul de la subsomption entre concepts, sur la base de leurs définitions respectives en termes de logique. D'un point de vue ensembliste, la subsomption est le calcul de la relation d'inclusion entre concepts.

[2] Dans l'optique d'une implantation relationnelle.

stricte[3], par ses vues parentes (sauf la vue racine, appelée vue minimale) et ses attributs et contraintes propres. Un P-type représente un ensemble homogène d'objets du monde réel, par exemple des personnes, des véhicules, des maladies. De tels ensembles sont par nature disjoints et un objet appartient à un P-type et un seul.

L'identification des vues d'un objet devant être permanente, aussi bien à la création de l'objet que lors de toute modification, une attention particulière a été portée à cette opération, qui est fortement optimisée, en particulier lors des mises à jour des objets. Le classement d'objet et la représentation des vues reposent sur une structuration de l'espace des objets qui est dérivée des propriétés logiques qui définissent les vues. L'espace des objets d'un P-type est ainsi partitionné en classes d'équivalence, appelées Eq-classes, qui regroupent des objets ayant le même comportement vis-à-vis de l'ensemble des vues de ce P-type : tous les objets de la même Eq-classe satisfont exactement le même ensemble de vues du P-type. Dès lors, l'unité de représentation et de raisonnement n'est plus l'objet élémentaire mais l'Eq-classe, qui à un instant donné représente l'ensemble des objets équivalents vis-à-vis du classement dans les vues. L'espace des objets structuré en Eq-classes est appelé l'Espace de Classement du P-type.

Nous présentons d'abord le modèle des P-types et le système Osiris, qui est le prototype qui implante le modèle. Dans cette partie nous introduisons également l'Espace de Classement, qui constitue la structuration conceptuelle de l'espace des objets. Nous présentons ensuite la représentation physique de l'Espace de classement et son usage pour l'indexation des objets. Nous montrons comment cette structuration de l'Espace de Classement peut servir de support à l'optimisation sémantique des requêtes et nous évoquons en conclusion le support à l'intégration de données que constitue cet espace.

## 2 Le modèle des P-types

Le système Osiris implante le modèle des P-types, qui a été initialement conçu dans l'optique d'un SGBD-BC privilégiant le partage d'information à travers les vues et assurant la vérification automatique de contraintes d'intégrité élaborées et le classement d'objets [18]. Nous présentons ci-dessous les notions principales du modèle des P-types.

### 2.1 Definitions.

**P-types**. L'ensemble des objets du domaine est organisé en sous-espaces disjoints, où chaque sous-espace, nommé P-type, représente les objets d'une même « famille ». Lors de la conception d'un Système d'Information, le choix des P-types est le premier

[3] Le modèle des P-types a été défini dans le paradigme des Types Abstraits Algébriques, qui a également été utilisé pour modéliser la notion d'objet dans les langages à objets. Il s'ensuit une certaine similarité entre une hiérarchie de spécialisation et une hiérarchie d'héritage. Toutefois, la première est strictement ensembliste alors que la seconde ne l'est qu'accidentellement, car son objectif est la réutilisation du code [4].

travail de l'analyste. Ce choix dépend des besoins de l'application. Ainsi, ETUDIANTS et EMPLOYES pourront constituer des espaces (P-types) disjoints ou au contraire être regroupés dans un même ensemble sous un P-type PERSONNE, dont ils constitueront alors des vues.

**Vues.** Un P-type est défini par une hiérarchie de vues dont la racine, appelée vue minimale, contient tous les objets du P-type. Une vue est définie par les vues qu'elle spécialise (sauf la vue minimale) et par ses attributs et contraintes propres.

**Attributs**. Un attribut a un nom (unique dans le P-type) et un type. Le type peut être prédéfini (INTEGER, REAL, BOOLEAN, CHARACTER, STRING), un P-TYPE (i.e., une référence à un P-type), ou une collection (set, list) de types prédéfinis ou de P-TYPES. Bien que les attributs puissent être définis dans n'importe quelle vue (et même dans plusieurs vues), dans cet article nous considèrerons pour simplifier que les attributs d'un P-type sont définis dans la vue minimale. Un attribut qui intervient dans au moins une contrainte est appelé un **attribut classificateur**.

**Contraintes.** Les contraintes définissant les vues sont des clauses de Horn dont les littéraux sont des prédicats élémentaires de domaine, de la forme Attr ∈ Domaine, où le domaine peut être un intervalle, (e.g., [10, 20]) ou un ensemble de valeurs énumérées (e.g., {vrai, faux}, {1, 3, 5, 7}, {bleu, rouge, brun, jaune}). Les prédicats de domaine peuvent être considérés comme une généralisation des *Simple Predicates*[4], qui sont utilisés dans les bases de données distribuées [15].

### 2.2 Exemple

Considérons un P-Type PERSON, avec quelques vues qui illustrent les définitions ci-dessus.

```
view PERSON                          -- Vue minimale du P-type PERSON
    attr  name: STRING;
      id: INT;
      age: INT in [0..120];          -- Contrainte de domaine : Age ∈ [0..120]
      sex: CHAR in {m, f};           -- Contrainte de domaine :Sex ∈ {m, f}
      salary : INT ≥ 0;              -- Contrainte de domaine : salary ∈ [0..SUP]
      owns: setof CAR;               -- Le P-type CAR est défini par ailleurs
      …
    (a1) age < 18 ➔ salary < 1200    -- Dépendance Inter-Attributs
end PERSON;

view MINOR: PERSON                   -- Spécialise la vue (minimale) PERSON
      age < 18                       -- Contrainte de domaine: age ∈ [0..18[
end MINOR;

view ADULT: PERSON                   -- Spécialise la vue (minimale) PERSON
      age ≥ 18                       -- Contrainte de domaine: age ∈ [18..120]
      salary ≥ 600
end ADULT;
```

[4] les *Simple Predicates* sont des prédicats de la forme Attr R Valeur, où R ∈ {=, <, ≤, >, ≥, ≠ }

```
view SENIOR: ADULT              -- Spécialise la vue ADULT
      age ≥ 65
     end SENIOR;
view MALE: PERSON               -- Spécialise la vue PERSON
      sex = m                   -- Contrainte de domaine: sex ∈ {m}
end MALE;
view EMPLOYEE: ADULT                    -- Specialise la vue ADULT
      salary ≥ 1200,00
end EMPLOYEE;
view CEO: EMPLOYEE              -- Spécialise la vue EMPLOYEE
    attr  manages : COMPANY ;   -- attribut propre à la vue CEO
      salary > 3000,00;
end CEO;
```

La forme la plus générale des contraintes (ou assertions) est celle des clauses de Horn (cf assertion (a1)). Nous en donnons un autre exemple ci-dessous, qui pourrait être associé à la définition de la vue minimale PERSON.

(a2) sex=m ➔ nbGrossesses = **undefined**

L'assertion (a2) est une contrainte d'intégrité qui exprime que le nombre de grossesses pour un individu masculin est *indéfini*. Osiris distingue les valeurs *indéfini* et *inconnu*, qui sont regroupées dans les bases de données classiques sous l'appellation *null. Indéfini* doit s'entendre ici au sens mathématique, de la même manière que la racine carrée d'un nombre négatif est indéfinie. En l'absence de cette distinction entre *indéfini* et *inconnu*, il est nécessaire (par exemple dans les bases relationnelles) de « fabriquer » un codage, du type nBgrossesses = 0, alors que la valeur *indéfini* traduit exactement la réalité de la situation. La valeur *indéfini* est utilisée par ailleurs dans le modèle des P-types, par exemple pour traduire qu'un attribut déclaré dans une vue « n'a pas de sens » hors de cette vue. C'est le cas par exemple de l'attribut *manages* déclaré dans la vue CEO, qui peut avoir la valeur *indéfini* pour un objet n'appartenant pas à la vue CEO.

### 2.3 L'Espace de Classement

L'Espace de Classement est une partition de l'espace des objets en classes d'équivalence (Eq-classes) telles que tous les objets d'une même Eq-classe appartiennent au même ensemble de vues.

**Definition** L'Espace de Classement d'un P-type est l'espace quotient de l'espace des objets relativement à la relation d'équivalence « *avoir la même valeur de vérité vis-à-vis de chacun des prédicats de domaine utilisés dans les contraintes de toutes les vues du P-type* »

**Sous-Domaines Stables (SDS)** Dans un P-type T, pour chaque attribut $\mathtt{Attr}_i$ soit P($\mathtt{Attr}_i$) l'ensemble des prédicats élémentaires sur $\mathtt{Attr}_i$ qui apparaissent dans les contraintes des vues de T. Un prédicat élémentaire est de la forme $\mathtt{Attr}_i \in D_{ik}$ où $D_{ik}$ est un sous-ensemble du domaine de définition $\Delta_i$ de $\mathtt{Attr}_i$.

Un prédicat élémentaire $Attr_i \in D_{ik}$ détermine une *partition* de $\Delta_i$ en deux éléments: $D_{ik}$ et $(\Delta_i - D_{ik})$. Le produit de toutes les partitions [22] définies par les prédicats de $P(Attr_i)$ constitue une partition de $\Delta i$ dont les *blocs* $d_{ij}$ sont appelés Sous-Domaines Stables (SDS).

Un SDS vérifie la **propriété de stabilité** *:* lorsque la valeur d'un attribut $Attr_i$ d'un objet $o_k$ varie à l'intérieur d'un SSD $d_{ij}$, l'objet $o_k$ continue de satisfaire exactement les mêmes prédicats de $P(Attr_i)$.

Etant donné l'ensemble des contraintes définies dans les vues du P-type PERSON, les produits des partitions pour les attributs *age*, *sex* et *salary* conduisent aux partitions suivantes de leur domaine:

age : $d_{11}=[0,18[$, $d_{12}=[18,65[$, $d_{13}=[65,120]$
sex: $d_{21}=\{f\}$, $d_{22}=\{m\}$
salary : $d_{31}=[0,600[$, $d_{32}=[600,1200[$, $d_{33}=[1200,3000[$, $d_{34}=[3000,SUP]$

où SUP désigne la plus grande valeur possible du domaine.

On note $SDS_{ATTR}$ l'ensemble de tous les SDS de l'attribut Attr. Dans l'exemple ci-dessus, les SDS des trois attributs classificateurs considérés sont :

$SSD_{age} = \{d11, d12, d13\}$
$SSD_{sex} = \{d21, d22\}$
$SSD_{salary} = \{d31, d32\ d33, d34\}$

**Eq-classe** La partition du domaine de chaque attribut d'un P-type est prolongée en une partition de l'espace des objets, qui constitue l'Espace de Classement du P-type. Chacun des n attributs classificateurs du P-type ayant été partitionné en SDS, l'Espace de Classement du P-type est un sous-ensemble du produit cartésien de ses SDS :

$SSD_1 * SSD_2 * \ldots * SSD_i * \ldots * SSD_n = \{<d_{1i}, d_{2j}, \ldots, d_{nk}> \mid d_{1i} \in SSD_1 \wedge \ldots \wedge d_{nk} \in SSD_n\}$

où $SDS_j$ représente l'ensemble des SDS de $Attr_j$, pour $j \in [1..n]$.

L'Espace de classement est un espace n-dimensionnel, où chaque Eq-classe est un hyper-cube représenté par un n-uplet de SDS. Pour des questions de représentation graphique, nous nous limitons à des espaces 3D. Ainsi, en considérant uniquement les trois attributs *age*, *sex* et *salary* l'Espace de Classement du P-Type PERSON est représenté dans la Fig. 1.

Les Eq-classes valides du P-type PERSON sont celles qui satisfont la vue minimale, représentées en gras sur la Fig.1. Par exemple, l'Eq-classe ($d_{13}$, $d_{22}$, $d_{34}$), qui, en particulier, contient l'objet (age=70, sex=f, salary = 5000) est valide, tandis que tout objet de l'Eq-classe ($d_{11}$, $d_{22}$, $d_{33}$) est invalide, car toute personne âgée de moins de 18 ans (age $\in d_{11}$) ne peut que satisfaire $d_{31}$= [0..600[ ou $d_{32}$= [600..1200[.

La propriété de stabilité d'un attribut peut être étendue à tout l'Espace de Classement.

**Propriété de stabilité d'une Eq-classe** : tous les objets d'une même Eq-classe ont la même valeur de vérité pour toutes les vues du P-Type.

**Corollaire 1** : lorsque la valeur d'un attribut d'un objet est modifiée tout en continuant d'appartenir au même SDS, l'objet continue de satisfaire les mêmes prédicats, donc les mêmes assertions, et en conséquence les mêmes vues.

**Corollaire2** : Lorsque plusieurs attributs d'un objet sont modifiés tout en se maintenant dans leur SDSs initiaux, l'objet continue de satisfaire les mêmes prédicats, donc les mêmes assertions, et en conséquence les mêmes vues.

Comme deux objets de la même Eq-classe satisfont les mêmes assertions, et en conséquence valident (et invalident) les même vues, il est possible de déterminer à priori les vues que les objets d'une Eq-classe satisfont. En conséquence, il est possible d'associer à chaque vue les Eq-classes qui la satisfont.

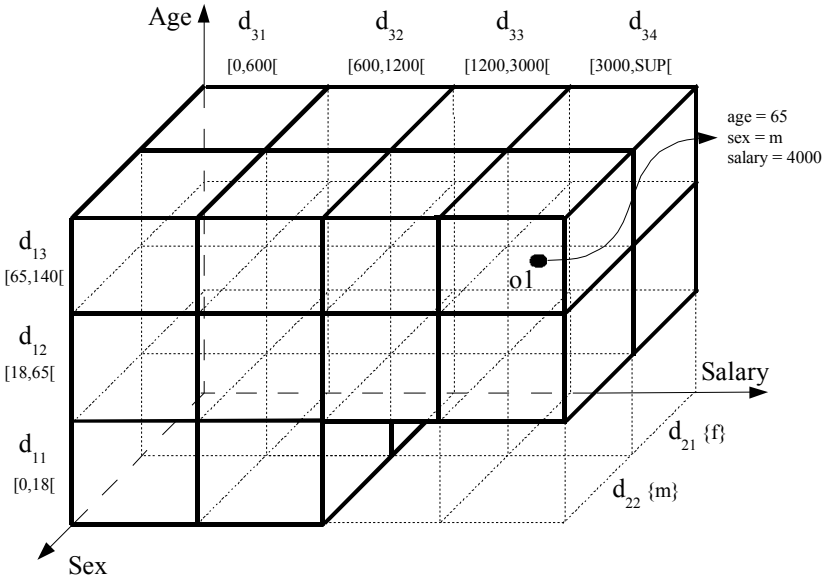


**Fig. 1.** Espace de Classement du P-Type PERSON

**Nombre d'Eq-classes**. Soit **n** le nombre d'attributs classificateurs du P-type **T**. Si chaque attribut a **p** SDS, le nombre d'Eq-classes de **T** est $\mathbf{p^n}$.

Ainsi, la taille de l'Espace de Classement est exponentielle au nombre d'attributs classificateurs, ce qui interdit en pratique sa représentation explicite. Le classement est réalisé par un réseau où les Eq-classes ne sont pas explicites, et les structures d'indexation ne représentent que les Eq-classes qui contiennent au moins un objet (cf §3.4), garantissant ainsi l'absence d'explosion combinatoire.

On notera que les DIAs (Dépendances Inter-Attributs) créent des « trous » dans l'Espace de Classement. Ainsi, les Eq-classes ($d_{11}$, $d_{21}$, $d_{33}$), ($d_{11}$, $d_{22}$, $d_{33}$), ($d_{11}$, $d_{21}$, $d_{34}$) et ($d_{11}$, $d_{22}$, $d_{34}$) sont exclues de la zone de validité de la vue minimale de PERSON et donc du P-type, ce qui traduit qu'aucune personne mineure ne peut avoir un salaire supérieur à 1200 (assertion (a1) dans PERSON). La « vérification » de la contrainte (a1) consiste simplement à s'assurer qu'aucun objet n'est situé dans les Eq-classes invalides de l'Espace de Classement. Ainsi, l'objet étant classé dans une Eq-classe, il faut et il suffit que cette Eq-classe appartienne aux Eq-classes valides du P-type pour que l'ensemble des contraintes d'intégrité du P-type soient satisfaites, sans qu'il soit nécessaire de les vérifier individuellement. Le classement de chaque instance suffit pour valider (ou invalider) les assertions du P-type.

## 3 Utilisations de l'Espace de Classement

L'Espace de Classement constitue un pivot du système Osiris, car il sous-tend de nombreux processus tels que le classement des objets, la classification des vues, la vérification des contraintes d'intégrité, la vérification de cohérence des contraintes elles-mêmes, l'indexation des objets, et l'optimisation sémantique des requêtes qui en découle. Nous présenterons succinctement chacun de ces processus, qui sont détaillés en [19][20].

### 3.1 Classement d'objet.

Le classement d'un objet consiste à déterminer ses vues valides et invalides, ainsi que ses vues potentielles lorsqu'il est incomplètement connu. Cette opération est réalisée en Osiris au moyen d'un automate à architecture connexionniste[5], que nous désignerons par *réseau de classement*. La couche d'entrée est constituée des sous-domaines stables des attributs classificateurs, et la couche de sortie par les vues du P-type. Pour un objet complètement connu, les activités possibles des cellules sont 1 et 0, représentant respectivement les SDS et les vues valides (1) et invalides (0) d'un objet donné. L'architecture de ce réseau est complètement déterminée par l'analyse du p-type, et il n'a pas besoin d'apprentissage. C'est donc un réseau exact, contrairement à la plupart des réseaux neuronaux qui sont utilisés pour le classement dans les bases de connaissances. Sa taille est $\sum_{i=1}^{N} n_i + (N+1)n_a + n_v$ pour un P-type ayant **N** attributs, $n_a$ assertions, $n_V$ vues, et où l'attribut i a $n_i$ sous-domaines stables. Une variante de ce réseau permet également la détermination des vues potentielles d'un objet incomplètement connu, et, lorsque des probabilités peuvent être associées aux sous-domaines stables, il calcule la probabilité qui en résulte pour les vues potentielles [2]. L'activité des cellules est alors dans l'intervalle réel [0,1].

### 3.2 Vérification de contraintes d'intégrité.

La vérification des contraintes d'intégrité, représentées par les assertions des vues, est un « sous-produit » du classement. La question de la vérification des contraintes d'intégrité se pose lors d'une affectation impérative d'un objet à une vue (par ex. **create** EMPLOYEE (…)), et plus tard d'une mise à jour de l'objet. Il est nécessaire de vérifier que les assertions de la vue EMPLOYEE sont vérifiées par les valeurs des attributs de l'objet créé. Comme tout objet est classé lors de sa création, ses vues sont déterminées. La vérification des contraintes d'une vue Vi revient donc à vérifier que cette vue appartient à l'ensemble des vues valides de l'objet. Lors d'une modification, si l'objet reste dans la même Eq-classe, ce qui est testé simplement en vérifiant que les

[5] Une architecture connexionniste, appelée aussi réseau de neurones, est constituée d'un graphe orienté de cellules où chaque cellule calcule une activité à partir des activités de ces cellules d'entrée et communique son activité aux cellules connectées en sortie.

attributs modifiés restent dans les mêmes sous-domaines stables, le reclassement est inutile, de même que la vérification des contraintes, puisque l'objet continue d'appartenir exactement aux mêmes vues. Lorsque certains attributs changent de sous-domaine stable, l'objet est à nouveau classé.

### 3.3 Classification des vues.

La classification des vues, i.e., la détermination de la relation de subsomption (d'inclusion) entre les vues, n'est pas réalisée explicitement. En effet, c'est une opération coûteuse, dont la complexité est NP dans le cas d'Osiris, comme pour la plupart des langages terminologiques [14]. Son intérêt est limité en Osiris puisque l'on dispose de la partie de l'espace de classement qui concerne les objets de la base.

### 3.4 Indexation des objets.

Une structure d'indexation, nommée ISD (Indexing Structure Descriptor), est définie pour chaque P-type [20]. Ses principaux champs sont :

- Un vecteur de Sous-Domaines Stables représentant l'Eq-classe indexant un ensemble d'objets. Deux SDS ont été ajoutées pour représenter les valeurs **inconnu** et **null** d'un attribut[6].
- Un vecteur de vues qui donne le statut (Valide, Invalide, Potentiel) de chaque vue du P-type pour l'ensemble des objets indexés par cet ISD.
- Une référence à l'ensemble des objets de l'ISD.
- Le nombre total d'objets indexés par cet ISD.

Un objet, même partiellement connu, appartient à un seul ISD. En effet, dans le cas où un attribut est inconnu, on considère que l'étendue de son SDS est le domaine de l'attribut dans le P-type, noté *****. On parle alors d'Eq-classe généralisée (ou simplement d'Eq-classe s'il n'y a pas d'ambiguïté). Dans cette situation, les ensembles d'Eq-classes correspondant à des ISD distincts peuvent ne pas être disjoints. Lorsqu'un attribut inconnu devient connu, l'objet change d'ISD (un nouvel ISD, subsumé par le précédent, est créé s'il n'existe pas déjà).

Lorsqu'un objet est créé, il est automatiquement classé. Ses sous-domaines stables et ses vues sont donc déterminés. Il est alors rangé dans un ISD, qui peut être créé si nécessaire. Lorsqu'un objet est modifié, il reste dans le même ISD si aucun attribut n'a changé de sous-domaine stable.

### 3.5 Optimisation sémantique des requêtes.

L'optimisation sémantique consiste à transformer une requête en une requête équivalente, c'est à dire une requête dont le résultat est le même que celui de la requête initiale pour tout état de la base de données, sur la base des informations

[6] On rappelle qu'au niveau du P-type, le domaine d'un attribut qui n'est pas défini dans la vue minimale contient valeur *null*.

sémantiques associées aux classes du schéma conceptuel de la base de données. Ces informations sont essentiellement les contraintes d'intégrité associées aux classes et aux vues. Dans [3] les auteurs décrivent les conditions de mise en œuvre de telles méthodes d'optimisation, fondées sur la classification de classes, vues et requêtes relativement à la relation de subsomption. Celle-ci doit donc être de complexité raisonnable, et ils proposent un ensemble 'maximal' d'opérateurs qui conserve une complexité polynomiale au calcul de la subsomption.

En Osiris, une requête est évaluée à l'intérieur d'un P-type donné. Elle est assimilée à une vue (dynamique) du P-type et n'est pas classée explicitement par rapport aux autres vues du P-type. Par contre, elle est réécrite en termes des sous-domaines stables des attributs qu'elle contient, ce qui revient à la situer dans l'espace de classement en termes d'Eq-classes.

En Osiris, les requêtes sont de la forme *(Ptype| Contexte | Condition)*, où *Ptype* identifie le P-type d'intérêt de la requête ; le *Contexte* est donné par une formule propositionnelle $\Phi(Vi)$ sur les vues, utilisant les connecteurs **and**, **or** et **not**. PERSON est un exemple de Ptype ; ADULT **or** MALE est un exemple de Contexte. La *Condition* exprime une condition logique sur les attributs du P-type et a la même forme que les assertions du langage Osiris. Toutefois, par simplicité, nous ne considérons pas ici le cas où la condition est une implication, et nous examinons uniquement les conditions qui sont sous la forme d'une conjonction de littéraux, constitués eux-mêmes de prédicats élémentaires ou de leur négation. Enfin, nous présentons uniquement le cas où le contexte n'est pas précisé, ce qui revient à considérer l'évaluation de la requête dans le P-type complet. Cette simplification permet d'oublier le vecteur de vues des ISD et donc de raisonner au niveau de leurs Eq-classes.

Soit le p-type PERSON (§2.2). Un exemple de requête est `(PERSON|age<15)`. En classant la requête dans la hiérarchie des vues, l'optimisation sémantique classique permettrait de restreindre la recherche des objets d'intérêt aux objets de la vue MINOR. En Osiris, la recherche du sous-espace englobant minimal d'une requête se fait de manière plus ciblée en utilisant les ISD. Lors de l'évaluation d'une requête, l'enjeu est donc de déterminer les propriétés logiques des ISD, qui permettent de les classer en trois catégories : 1) ceux dont les objets font, de manière certaine, partie de la réponse, 2) ceux dont il faut examiner les objets individuellement, et 3) ceux dont les objets ne font pas partie de la réponse.

Ainsi, en considérant la requête

$$Q = (PType \mid Attr_1 \in \delta_1 \wedge \ldots \wedge Attr_j \in \delta_j \wedge \ldots \wedge Attr_n \in \delta_n)$$

la méthodologie de principe pour l'optimisation sémantique est la suivante :

1. pour chaque attribut, détermination du plus petit nombre de SDS, $d_{ij}$, $d_{ik}$, … tel que $d_{ij} \cup d_{ik}, \cup \ldots \supseteq \delta i$, où $\delta i$ est le domaine du prédicat de domaine sur l'attribut $Attr_i$ dans la requête.
2. détermination du sous-espace englobant minimal de la requête. Pour cela on qualifie les ISD (ici restreints aux Eq-classes).
   Une Eq-classe ($d_{1i}$, … $d_{jk}$, …, $d_{nl}$) est :
   - **Valide** (noté **VS**) : $d_{1i} \subseteq \delta_1$, …, $d_{jk} \subseteq \delta_j$, … $d_{nl} \subseteq \delta_n$
   - **Invalide** : $d_{1i} \cap \delta_1 = \phi$ **or** …, $d_{jk} \cap \delta_j = \phi$ **or** … $d_{nl} \cap \delta_n = \phi$
   - **Potentielle** (noté **VP**) : ni Valide ni Invalide.

L'ensemble des Eq-classes Valides et Potentielles constitue le sous-espace englobant minimal de la requête.

3. extraction des objets des Eq-classes *actives* du SI subsumées par les Eq-classes VS et VP de la requête, notées Eq-classes propres de la requête.

Par exemple, soient les SDS de l'attribut age :

age : $d_{11}$ = [0, 18[, $d_{12}$ = [18, 65[, $d_{13}$ = [65,120]

et la requête : (PERSON | age < 70)

Les Eq-classes (ISD) d'intérêt de cette requête sont :

{($d_{12}$, *, *), ($d_{13}$, *, *)} où * désigne l'ensemble des SDS d'un attribut qui n'apparaît pas dans la requête.

La première est une Eq-classe (généralisée) VS et la seconde une Eq-classe (généralisée) VP. Lors de l'évaluation d'une telle requête, et en fonction des ISD présents dans le SI, les ISD actifs sont ceux se projetant sur l'une ou l'autre des Eq-classes ci-dessus (Eq-classes propres). Selon que l'Eq-classe propre est un ISD à valeur de vérité VS ou VP, les objets d'un tel ISD satisfont la requête ou doivent être filtrés par la condition de la requête. Ainsi, en considérant l'ISD <($d_{12}$, $d_{21}$, *) (…)…>, cet ISD est bien un ISD d'intérêt pour la requête car il se projette sur l'Eq-classe propre ($d_{12}$, *, *). De plus, comme cette Eq-classe est de type VS, tous les objets de l'ISD appartiennent à la réponse.

Nous considérons la situation où toutes les Eq-classes sont actives. Dans ce contexte, considérons la requête Q1 et l'espace de classement limité aux attributs *age* et *salary* de la requête (Fig. 2) :

Q1 : {PERSON | age > 25 and salary < 3000}

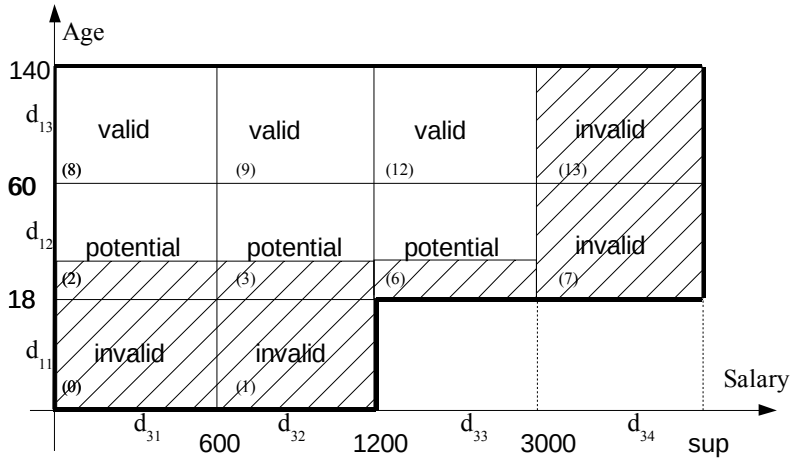


**Fig. 2.** Sous-espace englobant minimal de Q1

- les Eq-classes (8), (9) et (12) sont **Valides**, i.e., tous les objets de ces Eq-classes font partie du résultat de la requête.
- les Eq-classes (0), (1), (7) et (13) sont **Invalides**, i.e., aucun objet de ces Eq-classes ne fait partie du résultat de la requête.

- les Eq-classes (2), (3) et (6) sont **Potentielles**, i.e., les objets de ces Eq-classes doivent être testés individuellement pour décider s'ils font partie ou non du résultat de la requête.

Un travail est aujourd'hui en cours de réalisation pour indexer l'espace des ISD afin d'optimiser la recherche des ISD actifs d'intérêt pour une requête.

## 4 Conclusion et Perspectives

Conçu originellement pour la gestion des données, avec l'objectif de privilégier le rôle des utilisateurs et la vérification automatique des contraintes d'intégrité, le modèle des P-types offre également la fonction de classement des objets, caractéristique des bases de connaissances et toujours ignorée des SGBD classiques. Ce classement opère dans une hiérarchie de vues, qui peut être considérée comme une hiérarchie de classes dans le paradigme objet ou une hiérarchie de concepts dans le paradigme des ontologies. La nature ensembliste des vues, qui sont définies par des propriétés logiques, nous a conduits à considérer ce modèle dans la perspective des Logiques de Description [16] issues du langage KL-One[7]. Dans ce cadre, la vue minimale d'un P-type est un concept primitif[8] et les autres vues sont des concepts définis.

L'optimisation du classement, qui est fondamentale pour un usage dans les bases de données, où par nature un objet évolue dans le temps et peut changer de classe[9] (de vue dans le modèle des P-types). L'organisation de l'espace des objets autour des contraintes de domaine, qui constituent en pratique la forme de contraintes la plus utilisée, a permis de définir un nouvel espace dont les éléments sont les classes d'objets équivalents vis-à-vis des contraintes. C'est l'Espace de Classement, dont les éléments, appelés Eq-classes, deviennent les unités élémentaires de traitement en lieu et place des objets.

Les Eq-classes forment un hypercube dont les dimensions sont les attributs classificateurs du P-type. En pratique, comme sa taille est exponentielle au nombre des attributs classificateurs, il n'est jamais représenté explicitement. La représentation pratique de l'espace de classement se limite aux Eq-classes actives, c'est-à-dire qui contiennent au moins un objet, évitant ainsi l'explosion combinatoire, car la taille de l'Espace de Classement est alors bornée par le nombre d'objets de la base[10].

Cet hypercube n'est pas sans rappeler celui des entrepôts de données. Cette ressemblance n'est pas anodine, et l'implantation de vues semi-persistantes dans les

---

[7] On peut voir une certaine similarité entre l'émergence de KL-One, conçu dans un paradigme ensembliste et logique par réaction aux Frames qui étaient très « opératoires », et celle du modèle des P-types environ à la même époque. KL-One était dédié aux bases de connaissances tandis que les P-types répondaient à une problématique des bases de données, mais les deux modèles ont en commun de reposer sur une vision ensembliste des données.

[8] Mais tout concept primitif dans une DL ne peut être assimilé à un P-type.

[9] Les bases de données objet [8], en adoptant le modèle objet des langages de programmation, ont initialement fait l'impasse sur cette question du changement de classe d'un objet, qui est devenue par la suite un problème ouvert, et mal résolu à ce jour.

[10] Dans le pire des cas, où on aurait un seul objet par Eq-classe.

entrepôts de données est une des applications possibles du modèle de vues proposé en Osiris. En effet, dans les entrepôts de données, la maintenance des vues persistantes reste un problème majeur, et les requêtes qui ne sont pas optimisées sous la forme de vues persistantes peuvent avoir des temps de réponse catastrophiques. Le modèle d'indexation proposé par Osiris permet d'offrir un accès fortement optimisé aux objets d'une vue quelconque. Il reste à en faire une évaluation objective et la comparer aux performances offertes par les vues matérialisées dans les entrepôts de données.

La vérification des contraintes d'intégrité est un « effet de bord » du classement et ne nécessite pas de vérification spécifique des contraintes. En effet, l'Espace de Classement résulte d'une compilation des contraintes et l'identification de l'Eq-classe d'un objet suffit à déterminer si l'objet est valide pour le P-type.

Le modèle des P-types et son système de vues ont été appliqués à l'intégration de données. Deux thèses sont en cours, l'une sur l'intégration de données structurées et l'autre sur l'intégration de données semi-structurées de type XML. Dans les deux cas, la hiérarchie de vues d'un P-type joue le rôle dévolu aux ontologies dans les approches à base de médiateurs sémantiques, et l'Espace de Classement permet une semi-matérialisation du schéma global d'intégration [1][12].

Nous terminons en évoquant la question des données incomplètes (ou données manquantes). La réponse des SGBD classiques, qui consiste à attribuer la valeur **null** à un attribut inconnu[11], ne permet pas d'autre exploitation de ces données que leur élimination pour tout travail d'analyse de données. Dans l'univers des bases de connaissances et de l'aide à la décision, les travaux actuels s'orientent vers l'usage des réseaux Bayésiens ou l'introduction d'aspects probabilistes dans les approches classiques comme les arbres de décision [11].

En présence de valeurs d'attribut manquantes, le classement en Osiris détermine les vues Valides, Invalides et Potentielles. C'est le classement VIP. Lorsqu'un objet `o` n'est pas complètement valué, l'objet n'appartient pas à une Eq-classe, mais *potentiellement* à un certain nombre d'Eq-classes. Pour prendre en compte les objets incomplets, la méthodologie et le réseau de classement ont été adaptés [2][21]. Des travaux ont été engagés pour aboutir à une estimation probabiliste du classement [10] mais le problème reste la complexité de l'estimation des dépendances entre les attributs, qui reste nécessaire pour un classement probabiliste.

## References


1. Ahmad, H., Kermanshahani, S., Simonet, A. Simonet, M.: A Materialized Approach to the Integration of XML Documents: the OSIX System, ICOSE, the International Conference on Ontological and Semantic Engineering, Rome, Italy (2009)
2. Bassolet, C.G. : Approches Connexionnistes du Classement en Osiris. Vers un Classement Probabiliste. Thèse d'Informatique, Université Joseph Fourier (1998)
3. Buchheit, M., Jeusfeld, M. A., Nutt, W., Staudt, M.: Subsumption between queries to object-oriented database, Information Systems 19(1): pp 33-54 (1994)


[11] En outre, **null** désigne aussi bien une valeur *inconnue* (qui peut devenir connue) qu'une valeur *indéfinie* (qui n'a pas de sens).

4. Cardelli L., Wegner P.: On Understanding Types, Data Abstraction, and Polymorphism. in ACM Computer Survey, Vol 17( 4) (1985)
5. Bertino, E., Negri, M., Pelagatti, G., Sbattella, L.: Object-Oriented Query Languages : The Notion and the Issues, IEEE Trans. on Knowledge and Data Engineering, Vol. 4, No 3 (1992)
6. Borgida, A., Brachman, R.: Loading Data into Description Reasoners, in Proceedings of the ACM SIGMOD International Conference on Management of Data, pages 217-226, Washington, DC, June 1993.
7. Bresciani, P.: Querying Databases from Description Logics, KRDB'95, Workshop "Reasoning about Structured Objects : Knowledge Representation Meets Databases", in conjunction with KI'95, Bielefeld, Germany, Sept. 1995.
8. Cattell, R. G. G., Atwood, T., Duh, l J., Ferran, G., Loomis, M., Wade, D.: Object Database Standard : ODMG-93, Morgan Kaufmann Publishers (1994)
9. Date, C. J.: An Introduction to Database Systems, Addison-Wesley (1975)
10. Demongeot, J., Chaperon, T., Chouakria, A., Faraut, T., Simonet, A., Simonet, M.: Estimating joint probabilities in the context of probabilistic management of querying and integrity in a knowledge and data base, Int. Workshop on Conditional Independence Structures and Graphical Models, Fields Institute for Research in Mathematical Science, Toronto, Canada. (1999)
11. Hawarah, L., Simonet, A., Simonet, M.: Dealing with Missing Values in a Probabilistic Decision Tree during Classification. In Mining Complex Data, Studies in Computational Intelligence SCI 165, D. A. Zighed, S. Tsumoto, Z. W. Ras, H. Hacid (Eds), Springer, ISBN 978-3-540-88066-0, pp 55-74 (2008)
12. Kermanshahani, S.: Semi-Materialized Framework: a Hybrid Approach to Data Integration, *CSTST Student Workshop*, Paris (2008)
13. Minsky M., A framework for representing knowledge, in "Psychology of computer vision" P.H. Winston ed, Mc Graw Hill (1975)
14. Nebel, B.: Terminological Reasoning is inherently intractable, Artificial Intelligence, 43:235-249 (1990)
15. Ozsu, M.T., Valduriez, P.: Distributed Database Design: Fragmentation (Chap. 5.3), Principles of Distributed Database Design, Prentice Hall (1991)
16. Roger, M., Simonet, A., Simonet, M.: A Description Logic like model for a knowledge and data management system, DEXA 2000, LNCS, Springer, Heidelberg (2000)
17. Scholl, M. H., Laasch, C., Tresch, M.: Updatable Views in Object-Oriented Databases, Proc. 2nd DOOD conf., pp 187-198 (1991)
18. Sales-Simonet, A. : Types Abstraits et Bases de Données: formalisation du concept de partage et analyse statique de contraintes d'intégrité - Thèse Docteur Ingénieur, Université Scientifique et Médicale de Grenoble, France, Avril 1984.
19. Simonet, A., Simonet, M.: Objects with Views and Constraints : from Databases to Knowledge Bases, Object-Oriented Information Systems OOIS'94, D. Patel, Y. Sun and S. Patel eds, London, Springer Verlag, pp 182-197 (1994)
20. Simonet, A., Simonet, M. : Classement d'instance et Evaluation des Requêtes en Osiris, in BDA'96 : Bases de Données Avancées, pp 273-288 (1996)
21. Simonet, A., Simonet, M., Bassolet, C. G., Delannoy, X., Hamadi, R.: Static Classification Schemes for an Object System, FLAIRS-98, 11th Int. FLorida Artificial Intelligence Research Society Conference, AAAI Press, pp 254-258 (1998)
22. Stanat, D., McAllister, D.: Discrete Mathematics in Computer Science, Prentice Hall (1977)