



Université de
Sherbrooke

Théorie et modèles relationnels

Fondements

TMR_02

Christina KHNAISSER (christina.khnaisser@usherbrooke.ca)

Luc LAVOIE (luc.lavoie@usherbrooke.ca)

(les auteurs sont cités en ordre alphabétique nominal)

—

Scriptorium/Scriptorium/TMR_02-Fondements, version 0.1.5.a, en date du 2025-01-08

— document de travail, ne pas citer —

Sommaire

Le présent module a été rédigé dans le cadre de l'exploration du thème «Modélisation, conception et exploitation de données» (MCED) par des membres du CoFELI. Le module présente l'essentiel de la théorie relationnelle et en fixe la notation pour les autres modules du thème. Il découle en partie des travaux entrepris au sein des groupes de recherche Μητις et GRIIS. Il correspond à des connaissances usuellement couvertes au sein d'une formation universitaire en informatique (B. Sc. nord-américain, licence ou master européen) et de celles proposées par certaines écoles d'ingénieur. Ce module est destiné aux personnes étudiant les disciplines de l'informatique, de l'informatique appliquée et du génie logiciel. Nous espérons toutefois qu'il pourra être utile à toute personne curieuse d'approprier ce champ de connaissance.

Mise en garde

Le présent document est en cours d'élaboration; en conséquence, il est incomplet et peut contenir des erreurs.

Historique

diffusion	resp.	description
2025-01-08	LL	Déplacement de la section opérationnalisation dans le module TMR_03.
2024-09-06	LL	Revue
2024-08-23	CK	Transfert matériel TMR_01-Fondements.ppt
2023-10-01	LL	Adaptation au cadre du CoFELI.
2023-09-07	LL	Intégration au CoFELI et encodage à l'aide d'Asciidoc.
2023-09-06	LL	Ajout des critères de réfutabilité et d'éthique. Correction de coquilles.
2023-01-11	LL	Corrections stylistiques.
2023-01-10	LL	Corrections stylistiques — présentation des opérateurs d'agrégation.
2022-01-10	LL	Ajout de la base et de son constructeur.
2021-10-24	LL	Harmonisation avec la présentation BD010 préparée pour Douala.
2021-03-19	LL	Introduction des dénominations « type primitif » et « sous-type » ; distinction de l'union disjointe et de la réunion (ou somme) disjointe.
2021-01-26	LL	Systématisation de la présentation, correction de coquilles, éclaircissements divers, introduction des traductions SQL des opérateurs relationnels primaires.
2020-09-20	LL	Début de présentation de la notation relative aux couples et aux ensembles
2019-07-25	LL	Correction des coquilles signalées par Maxime Routhier.
2019-06-23	LL	Correction de coquilles.
2019-05-01	LL	Récupération de notes diverses.

Table des matières

Introduction	4
1. Présentation	5
1.1. L'intuition à l'origine de la théorie relationnelle	5
1.2. Pourquoi le modèle relationnel ?	5
2. Structure relationnelle	6
2.1. Définition	6
2.2. Logique	7
2.3. Ensemble	8
2.4. Type	10
2.5. Attribut	10
2.6. Tuple	11
2.7. Relation	12
2.8. Base	13
3. Opérateurs relationnels	14
3.1. Opérateurs primaires	15
3.2. Opérateurs usuels	19
3.3. Opérateurs spécialisés	23
3.4. Opérateurs de regroupement	23
3.5. Opérateurs d'agrégation	23
3.6. Opérateurs de graphes	23
3.7. Opérateurs déclassés	23
Conclusion	24
Définitions	25
Sigles	28
Références	29

Introduction

Le présent document a pour but de présenter une synthèse de la théorie relationnelle proposée par Edgar F. Codd et développée grâce, notamment, aux contributions de Christopher J. Date, Raymond F. Boyce, Nikos A. Lorentzos et Jeffrey D. Ullman.

La présentation repose sur des bases minimales :

- la logique du premier ordre,
- la théorie des ensembles (dont la théorie des types),
- l'arithmétique entière,
- les langages rationnels.

Évolution du document

Le présent document tire son origine de l'expérience d'enseignement des auteurs. Cette présentation n'a cessé d'évoluer depuis grâce aux étudiants et auxiliaires d'enseignement qui ont participé aux cours depuis. La rédaction du document a commencé en août 2012. La première version du document a été établie sur les bases suivantes :

- le matériel pédagogique développé par l'auteur dans le cadre de formations relatives aux bases de données assurées entre 1983 et 2024 au Québec, en France, en Tunisie, en Suisse, au Maroc, au Liban et au Cameroun ;
- des échanges avec de nombreux étudiants et collaborateurs, plus particulièrement Zouhir Abouaddaoui, Samuel Dussault, Marc Frappier, Aurélie Ottavi et Maxime Routhier ;
- les différents travaux publiés par Codd, Darwen, Date, Delobel, Elmasri, Lorentzos, Navathe, Snodgrass et Ullman.

Contenu des sections

- La section 1 présente le contexte historique et scientifique ayant mené à l'élaboration de la théorie relationnelle et de ses modèles.
- La section 2 présente l'essence de la théorie relationnelle et le modèle qui sera utilisé tout au long de la présentation.
- La section 3 présente les opérateurs relationnels tant élémentaires que dérivés et composites.

Sections à venir (en cours de rédaction, mais non encore disponibles)

- La section 4 explicite le lien entre la théorie relationnelle et la logique du premier ordre en développant les correspondances entre relation et prédicat, tuple et proposition.
- La section 5 intègre le constructeur de type intervalle, les opérateurs de Allen et les opérateurs relationnels qui en tirent parti (tels que proposés par Darwen, Date et Lorentzos).
- La section 6 présente et compare un sous-ensemble minimal de trois langages de programmation relationnels.

1. Présentation

La logique et la déduction rationnelle sont les outils privilégiés par la science pour encadrer notre perception du monde, le penser et l'expliquer. Thalès, Pythagore, Platon, Aristote, Archimède, Pappus, Descartes, Tarski, Russell, Whitehead, Wittgenstein, Gödel et Turing sont quelques-uns des penseurs qui ont contribué à développer notre compréhension de ces outils et à définir ce que sont une théorie, un modèle et un système axiomatique.

Plus récemment, Floyd, Codd, Hoare, Dijkstra, Gries, Date, Ullman et d'autres, ont contribué à rendre effectivement calculable un sous-ensemble significatif des propriétés des systèmes construits sur les bases établies par leurs prédécesseurs.

Ces contributions sont presque entièrement intégrées dans la théorie, les modèles et les systèmes relationnels que nous décrirons ici et qui sont le fondement des systèmes de gestion de bases de données (SGBD) les plus performants développés à ce jour, à savoir les SGBD relationnels (SGBDR).

1.1. L'intuition à l'origine de la théorie relationnelle

- Le point de départ : l'équivalence entre fait, proposition et tuple.
 - Des opérations : la logique aristotélicienne bivaluée (dite logique classique).
 - Une limite, source de puissance : la vision encapsulée dans un monde fermé.
- Une généralisation : l'équivalence entre catégorie, prédicat et relation.
 - Des opérations : la théorie des ensembles.
- Le couplage la théorie relationnelle à la théorie des types.
 - Les outils : le générateur de type de relation, le type de relation, la relation, la variable de relation et la fonction de relation.
- L'automatisation du calcul.
 - La dualité donnée-calcul de Von Neuman et son corolaire : la relation est représentable tant par un ensemble de tuples (données) que par une expression (calcul).
 - Une intuition : permettre les deux façons d'exprimer la relation et laisser l'engin choisir la représentation ; justification de la dualité entre relation « matérialisée » et relation « virtuelle ».

1.2. Pourquoi le modèle relationnel ?

Lorsque Codd jette les bases de la théorie relationnelle en 1969 et propose un premier modèle opérationnalisable en 1970, il s'inscrit en continuité d'une histoire déjà longue dont voici quelques jalons choisis :

- Les fonctionnaires babyloniens (XVIII^e siècle avant notre ère).
- Ératosthène (III^e siècle avant notre ère) ;
- Les ingénieurs romains (I^{er} siècle) ;
- Cardan et Kepler (XVI^e siècle) ;

Ce n'est toutefois pas cette histoire qui permit à la théorie et au modèle de Codd de s'imposer rapidement à partir des années 1970 et de rester dominant jusqu'à nos jours, mais

- la capacité d'opérationnaliser des modèles de données tant simples que très complexes,
- la capacité de formuler la description d'un raisonnement, fondé sur la logique du premier ordre, dont le calcul est automatisable et
- la capacité d'évaluer objectivement plusieurs critères d'adéquation d'une solution en regard d'un problème.

Formalisation

- définissant l'équivalence des « faits » (proposition, variable, prédicat);
- définissant l'équivalence des opérateurs (relationnels et logiques);
- définissant les notions d'équivalence et d'égalité;
- utilisant la notion d'ensemble;
- utilisant des attributs dénotationnels plutôt que positionnels;
- soutenant une algèbre complète au sens de Turing.

Adéquation

Des huit principaux critères d'adéquation (voir TMR_01), sept sont couverts de façon utile, voire complète, par la théorie relationnelle, ses bases et les théories informatiques complémentaires (telle que la théorie de la complexité). Le huitième critère, l'éthique (la conformité aux principes et règles de conduite d'une société humaine de référence), y échappe cependant.

Modèles et méta-modèles

On remarque que les modèles issus de la théorie relationnelle permettent la définition de modèles de données. Ce sont donc ce qu'il est convenu d'appeler des méta-modèles. De plus, ils ont souvent la particularité de pouvoir se décrire eux-mêmes! Cette propriété incite donc la plupart des concepteurs de SBGB à y inclure une base de données, appelée catalogue, décrivant les bases de données gérées par le SGBD (dont le catalogue lui-même). Cette approche a l'avantage de permettre l'utilisation des opérateurs relationnels (et de tout l'environnement d'exploitation) lors de la gestion et de l'évolution des bases de données du SGBD.

2. Structure relationnelle

2.1. Définition

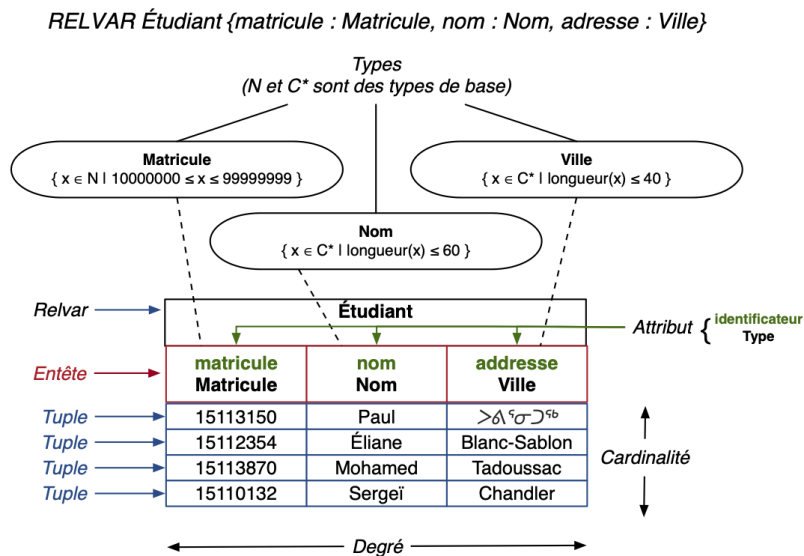
2.1.1. Définition informelle

La représentation naïve

Tableau 1. Étudiant

matricule	nom	ville
15113150	Paul	>Δ ^ς σϯ ^ς ᵇ
15112354	Éliane	Blanc-Sablon
15113870	Mohamed	Tadoussac
15110132	Sergeï	Chandler

Une représentation plus élaborée



Toutefois, aucune de ces représentations ne permet de transformer les données en information puisqu'il y manque le prédicat de définition !

2.1.2. Définition formelle

La théorie relationnelle est construite sur la base de la logique du premier ordre et de la théorie des ensembles.

En pratique, il faut aussi ajouter un modèle de typage.

Finalement, il est également utile d'y ajouter l'arithmétique entière et les langages rationnels afin de rendre compte de certaines propriétés des nombres entiers et des textes considérés comme types primitifs avec les booléens.

Un modèle relationnel minimal pourrait ne comprendre qu'un seul type primitif, les booléens. Son expressivité théorique serait la même que celui comprenant les entiers et les booléens. Pour utiliser ces derniers, il « suffirait » de les définir en termes de relations et de leur associer les opérateurs requis à l'aide de fonctions.

Dans les systèmes appliqués, il est fréquent d'ajouter d'autres types primitifs ou prédéfinis tels que :

- estampille temporelle [TIMESTAMP],
- nombres rationnels décimaux [DECIMAL],
- nombres rationnels « flottants » [DOUBLE PRECISION].

Il est également possible d'inclure des constructeurs de types tels que :

- énumération [ENUM],
- tableau [ARRAY].

2.2. Logique

Nous utiliserons les opérateurs suivants pour les expressions logiques (adaptation libre de <https://fr.wikipedia.org/wiki/Portail:Logique>, consulté les 2021-10-24 et 2024-04-07) :

Tableau 2. Opérateurs de la logique du premier ordre

Symbole	Spécification
$\neg A$	Négation de A. Non A.
$A \wedge B$	Conjonction. A et B.
$A \vee B$	Disjonction (inclusive). A ou B.
$A \Rightarrow B$	Implication. Si A alors B.
$A \Leftrightarrow B$	Équivalence. A est équivalent à B ; on dit aussi : A si et seulement si B.
$\Gamma \vdash A$	Déduction. De l'ensemble de formules Γ on déduit A.
$M \models A$	Réalisabilité. M réalise A, on dit aussi que M « force » A.
$M \models A$	Modélisation. M est un modèle de A ; on dit aussi A est vraie dans M.
$\vdash A$	Théorème. Notion syntaxique.
$\models A$	Tautologie. Notion sémantique.

Au sujet de l'équivalence, les symboles \equiv et \equiv sont également souvent utilisés parfois associés à des nuances dont non traitées dans le présent document.

2.3. Ensemble

Dans le cadre de la théorie relationnelle, les éléments d'un ensemble sont toujours contraints par un type associé à l'ensemble lui-même. Ce type est désigné comme le « type de référence » de l'ensemble et noté $typeref(e)$ où e est un ensemble.

Cette association peut être explicite (par voie de déclaration) ou déduite grâce au contexte.

2.3.1. Cas général

Les ensembles sont simples, ils ne possèdent qu'un constructeur et une propriété.

Le nombre d'opérateurs est toutefois considérable.

Nous n'en présenterons que quelques-uns.

Dénotation des valeurs et propriétés

$\{\}$

l'ensemble vide.

$\{z\}$

l'ensemble comprenant une seule valeur dénotée par l'expression z
(synonyme : singleton).

$\{z_1, z_2, \dots, z_n\}$

l'ensemble comprenant les seules valeurs dénotées par les expressions z_i .

Le point-virgule « ; » peut être utilisé indifféremment en lieu et place de la virgule « , ».

$\#e$

la cardinalité de l'ensemble e , soit le nombre d'éléments qu'il contient..

Opérateurs usuels

$a = b$

égalité de a et b .

$a \neq b$

inégalité de a et b .

$z \in e$

l'appartenance d'un élément z à un ensemble e (aussi noté $e \ni z$).

$$e_1 \subseteq e_2$$

l'inclusion, e_1 est un sous-ensemble de e_2 (aussi noté $e_2 \supseteq e_1$).

$$e_1 \subset e_2$$

l'inclusion stricte, e_1 est un sous-ensemble strict de e_2 (aussi noté $e_2 \supset e_1$).

$$e_1 - e_2$$

la différence de e_2 moins e_1 (non commutatif).

$$e_1 \cup e_2$$

l'union de e_1 et e_2 (commutatif).

$$e_1 \cap e_2$$

l'intersection de e_1 et e_2 (commutatif).

$$e_1 \times e_2$$

le produit cartésien de e_1 par e_2 (non commutatif).

Inclusion de la négation (liste très partielle)

$$z \notin e$$

la non-appartenance d'un élément z à un ensemble e (aussi noté $e \not\supset z$).

$$e_1 \not\subset e_2$$

e_1 n'est pas un sous-ensemble strict de e_2 (aussi noté $e_2 \not\supset e_1$).

Raccourcis

Dans la mesure où tous les ensembles ont un type de référence, l'opérateur « \backslash » permet de calculer le complément d'un sous-ensemble par rapport à son type de référence.

$$\backslash e \triangleq \text{typeref}(e) - e$$

où e est une expression dont la valeur est un ensemble typé.

Notes

- En Tutorial D, cet opérateur n'est utilisable que dans un contexte de dénotation explicite des valeurs, ainsi $\{ \text{ALL BUT } e_1, \dots, e_n \}$ dénote $\backslash \{ e_1, \dots, e_n \}$. On aurait souhaité, la syntaxe plus générale suivante ALL BUT exp , où exp dénote toute expression ensembliste typée.
- En Discipulus, la syntaxe générale est utilisée et les expressions d'ensemble (en particulier d'ensembles d'identifiants d'attributs) sont dénotables.

2.3.2. Couple

Le couple (paire) est la formation d'une nouvelle entité (ou élément) à partir de deux autres.

Il serait possible de reformuler le couple en termes d'ensemble, son utilisation au titre de construction propre relève donc uniquement de la commodité.

Dénotation des valeurs et propriétés

$$(x; y)$$

pour x et y des expressions dénotant des valeurs appropriées.

Note

- La virgule « $,$ » est souvent utilisée en lieu et place du point-virgule « $;$ » lorsqu'elle n'entraîne pas d'ambiguïté (e.a. en présence de nombres rationnels représentés en notation fractionnaire décimale).

Opérateurs usuels

$$@1(x; y) = x$$

l'élément gauche du couple;

$@2(x; y) = y$

l'élément droit du couple.

2.4. Type

Si la théorie relationnelle peut s'accommoder de plusieurs modèles de typage, certaines exigences doivent être respectées.

- Malheureusement, il n'y a pas de consensus relativement à ces exigences (notamment, celles du comité ISO 9075 diffèrent de celles énoncées par Codd et développées par Date et Darwen).
- Nous nous en tiendrons donc ici à un modèle de typage très simple, également acceptable en regard des jeux d'exigences les plus courants.

Une donnée est une valeur associée à deux représentations :

- l'une interne, apte à être traitée par ordinateur ;
- l'autre externe, apte à être utilisée au sein d'un langage formel (lui-même apte à être traité « commodément » par un ordinateur).

Une représentation est une suite de signaux (un signal est un phénomène physique mesurable, donc suffisamment stable pour être mesuré).

Dans le cadre de la théorie minimaliste de typage utilisée dans le présent document, un type est soit un type de base, soit un sous-type.

- Un type de base (appelé aussi type racine ou domaine) est un ensemble fini de valeurs propres (c'est-à-dire que les valeurs d'un type de base n'appartiennent à aucun autre type de base). Une valeur est donc un élément d'un type de base.
- Un sous-type (appelé aussi type dérivé) est un sous-ensemble d'un type, sous-ensemble déterminé par une contrainte explicite (qui restreint les valeurs acceptées dans le sous-ensemble).



La finitude des types suscite encore de nombreux débats.

À telle enseigne que Date lui-même l'a retiré de sa définition depuis 2016, moins par conviction selon nous, mais plutôt pour éviter un débat peu productif.

N'ayant pas sa notoriété ni son influence, nous pouvons nous permettre de la maintenir, et de l'utiliser, dans le présent document.

S'il est vrai que la structure mathématique, et plus particulièrement algébrique du modèle relationnel, ne repose pas sur la finitude des types, son application pratique, elle, repose sur cette hypothèse — du moins, tant que nous ne bénéficierons pas de machine de calcul (ordinateur) dotée d'une mémoire infinie.

2.5. Attribut

Un attribut est un couple formé d'un identifiant a et d'un type D , noté $a:D$.

Par abus de langage, lorsque le contexte le permet, il est usuel de désigner l'attribut par son seul identifiant ; ainsi peut-on écrire « l'attribut a ».



C'est-à-dire que le contexte permet de déterminer de façon univoque et non ambiguë le type associé à l'identifiant au sein de l'attribut.



La mise en page des expressions formelles des trois prochaines sous-sections (tuple, relation, base) varie d'une sous-section à l'autre. C'est volontaire. Nous y expérimentons trois mécanismes différents mis à disposition par AsciiDoc pour ce faire.

Nous désirons recueillir ainsi les commentaires de nos lecteurs, afin de choisir ce qui convient le mieux. Vos commentaires sont donc bienvenus et attendus!

2.6. Tuple

Un tuple, en tant qu'objet de la théorie relationnelle, est une « composition » d'attributs.

En regard de la théorie des types, il s'agit d'un constructeur (de types non scalaires) applicable à un ensemble d'attributs.

Soit n un entier naturel,

soit n identifiants **distincts** dénotés $a_i \mid 1 \leq i \leq n$,

soit n types (pas nécessairement distincts) dénotés $D_i \mid 1 \leq i \leq n$,

soit n valeurs (pas nécessairement distinctes) dénotées $v_i \mid (a_i \in D_i) \wedge (1 \leq i \leq n)$ alors

un tuple t est défini comme suit :

$$t \triangleq (\{a_i:D_i \mid 1 \leq i \leq n\}; \{(a_i, v_i) \mid 1 \leq i \leq n\})$$

ou, de façon équivalente, en extension :

$$t \triangleq (\{a_1:D_1, \dots, a_i:D_i, \dots, a_n:D_n\}; \{(a_1, v_1), \dots, (a_i, v_i), \dots, (a_n, v_n)\})$$

Sont également définis

$deg(t) = n$	degré de t
$id(t) = \{a_i \mid 1 \leq i \leq n\}$	ensemble des identifiants d'attributs de t
$def(t) = \{a_i:D_i \mid 1 \leq i \leq n\}$	entête de t
$val(t) = \{(a_i, v_i) \mid 1 \leq i \leq n\}$	valeur de t
$ANTE \ a \in id(t) : def(t, a) = D \mid a:D \in def(t)$	type de l'attribut a_i de t
$ANTE \ a \in id(t) : val(t, a) = v \mid (a, v) \in val(t)$	valeur de l'attribut a_i de t

Observation

Pourquoi autoriser $deg(t) = 0$?

- Le cas $deg(t) = 0$ correspond à un tuple remarquable $(\{\}; \{\})$ souvent désigné par l'identifiant t_0 .

Exercice

Expliquer pourquoi ce tuple est remarquable... et essentiel !

Autres dénotations fréquentes

Parmi les plus fréquentes :

- dénotation des tuples fondée sur l'ordre d'énumération des attributs (les identifiants d'attributs et leurs types étant déterminés par ailleurs):
 - $\langle v_1, v_2, \dots, v_n \rangle$
- dénotation des attributs au sein d'un tuple, $[.Math]\#val(t, a)$:
 - $t.a$
 - $a(t)$
 - $t(a)$
 - $a \text{ FROM } t$

Dénotation des constructeurs de types et de valeurs

Il est important, en pratique et du point de vue du génie logiciel, de différencier syntaxiquement le constructeur de types du constructeur de valeurs, ce que Tutorial D ne fait pas de façon assez explicite, nous semble-t-il.

Deux solutions sont généralement utilisées,

- des mots-clés différents, par exemple **TUPLE** et **TUP** ;
- un seul mot-clé et des parenthèses distinctives, par exemple **TUPLE{}** et **TUPLE[]**.

Discipulus a adopté cette dernière solution... pour le moment. Ainsi,

- le type tuple sans attribut est-il dénoté par **TUPLE {}** ;
- et la (seule) valeur (possible) d'un tuple sans attribut par **TUPLE []**.

2.7. Relation

Une relation, en tant qu'objet de la théorie relationnelle, est un ensemble de tuples.

En regard de la théorie des types, il s'agit d'un constructeur (de types non scalaires) applicable à un ensemble de tuples de même type.

Soit a_i des identifiants distincts,

soit D_j des types,

soit t_k des tuples alors

une relation R est définie comme suit :

$$R \triangleq (\{a_1:D_1, \dots, a_i:D_i, \dots, a_n:D_n\}; \{t_1, \dots, t_k, \dots, t_m\})$$

avec

$$\forall 1 \leq k \leq \text{card}(R) : \text{def}(R) = \text{def}(t_k)$$

Sont également définis

$$\text{deg}(R) = n$$

le degré de R

$$\text{card}(R) = m$$

la cardinalité de R

$$\text{id}(R) = \{a_1, \dots, a_i, \dots, a_n\}$$

l'ensemble des identifiants d'attributs de R

$$\text{val}(R) = \{t_1, \dots, t_k, \dots, t_m\}$$

la valeur de R

$$\text{def}(R) = \{a_1:D_1, \dots, a_i:D_i, \dots, a_n:D_n\}$$

l'entête de R

$$\text{def}(R, a_i) = D_i$$

le type de l'attribut a_i de R

Quatre tuples (ayant le même entête)

matricule: Matricule	nom: Nom	adresse: Ville	t1
matricule: 15113150	nom: Paul	adresse: >Δ ^ς σ ^ς ᵇ	
matricule: Matricule	nom: Nom	adresse: Ville	t2
matricule: 15112354	nom: Éliane	adresse: Blanc-Sablon	
matricule: Matricule	nom: Nom	adresse: Ville	t3
matricule: 15113870	nom: Mohamed	adresse: Tadoussac	
matricule: Matricule	nom: Nom	adresse: Ville	t4
matricule: 15110132	nom: Sergeï	adresse: Chandler	

Une relation comprenant quatre tuples

matricule: Matricule	nom: Nom	adresse: Ville	
matricule: Matricule	nom: Nom	adresse: Ville	t1
matricule: 15113150	nom: Paul	adresse: >Δ ^ς σ ^ς ᵇ	
matricule: Matricule	nom: Nom	adresse: Ville	t2
matricule: 15112354	nom: Éliane	adresse: Blanc-Sablon	
matricule: Matricule	nom: Nom	adresse: Ville	t3
matricule: 15113870	nom: Mohamed	adresse: Tadoussac	
matricule: Matricule	nom: Nom	adresse: Ville	t4
matricule: 15110132	nom: Sergeï	adresse: Chandler	

La représentation compacte usuelle de cette même relation

matricule: Matricule	nom: Nom	adresse: Ville
15113150	Paul	>Δ ^ς σ ^ς ᵇ
15112354	Éliane	Blanc-Sablon
15113870	Mohamed	Tadoussac
15110132	Sergeï	Chandler

Prédicat et relation

Un prédicat peut être défini de deux façons et, corolairement, une relation aussi :

- par énumération (l'ensemble de tous les énoncés vrais et eux seuls);
- par compréhension (la caractérisation nécessaire et suffisante des relations entre les variables).

Observation

Pourquoi autoriser $\deg(R) = 0$?

Le cas $\deg(R) = 0$ correspond à deux (valeurs de) relations remarquables :

- $(\{\}; \{\})$
- $(\{\}; t_0)$

Elles sont très importantes, comme le zéro et le un pour les entiers !

Exercice

Expliquer pourquoi !

Dénotation des constructeurs de types et de valeurs

Il est important, en pratique et du point de vue du génie logiciel, de différencier syntaxiquement le constructeur de types du constructeur de valeurs, ce que Tutorial D ne fait pas de façon assez explicite, nous semble-t-il.

Attendu la solution retenue par Discipulus (voir section précédente), les (deux seules) valeurs de relations sans attributs sont représentées par

- **RELATION** [] et
- **RELATION** [TUPLE []].

Cela ne rappelle-t-il pas quelque chose aux Bourbakistes parmi vous ?

2.8. Base

Une base, en tant qu'objet de la théorie relationnelle, est une « composition » de relations.

En regard de la théorie des types, il s'agit d'un constructeur (de types non scalaires) applicable à un ensemble de relations.

Soit v_i des identifiants distincts,

soit D_i des types de relation,

soit r_i des (valeurs de) relations alors

une base (banque) B est définie comme suit :

$$B \triangleq (\{v_1:D_1, \dots, v_i:D_i, \dots, v_n:D_n\}; \{r_1, \dots, r_i, \dots, r_n\})$$

avec

$$\forall 1 \leq i \leq \text{card}(B) : \text{def}(B, v_i) = \text{def}(r_i)$$

Sont également définis

$$\text{deg}(B) = n$$

degré de B

$$\text{id}(B) = \{v_1, \dots, v_i, \dots, v_n\}$$

ensemble des identifiants de B

$$\text{val}(B) = \{r_1, \dots, r_i, \dots, r_n\}$$

valeur de B

$$\text{def}(B) = \{v_1:D_1, \dots, v_i:D_i, \dots, v_n:D_n\}$$

entête de B

$$\text{def}(B, a_i) = D_i$$

type de a_i de B

3. Opérateurs relationnels

L'algèbre relationnelle est souvent présentée à l'aide de six opérateurs relationnels primaires :

- **renommage**, $R \rho a:b$, la relation comprenant tous les tuples formés à partir d'un tuple de R dont l'attribut de nom a est remplacé par un attribut de nom b de même valeur, et rien d'autre ;
- **restriction**, $R \sigma c$, la relation comprenant tous les tuples de R satisfaisant la condition c , et rien d'autre ;
- **projection**, $R \pi x$, la relation comprenant tous les tuples formés à partir d'un tuple de R dont seuls les attributs dont le nom est parmi x ont été conservés, et rien d'autre ;
- **jointure**, $R \bowtie S$, la relation comprenant tous les tuples formés des attributs d'un tuple de R et de ceux d'un tuple de S dont les attributs de même nom sont de même valeur, et rien d'autre ;
- **union**, $R \cup S$, la relation comprenant tous les tuples de R et tous les tuples de S , et rien d'autre ;
- **différence**, $R - S$, la relation comprenant tous les tuples de R qui ne sont pas dans S , et rien d'autre.

Note sur le renommage

Le statut de l'opérateur de renommage est encore discuté au sein de la communauté scientifique.

1. Il est possible de se passer de l'opérateur de renommage si on intègre une structure de catalogue à la théorie relationnelle, puisque le renommage peut alors être exprimé à l'aide des autres opérateurs relationnels appliqués aux variables de relation appropriées du catalogue.
2. L'opérateur de renommage découle d'un principe plus général, le principe de substitution. En ce sens, il n'appartient pas à la théorie relationnelle en propre, mais à la formalisation de toute algèbre.

Toutefois, le prix à payer afin d'omettre l'opérateur de renommage est une complexification de la formalisation de la théorie et, corolairement, des modèles qui en découlent et donc des expressions relationnelles en regard de ceux-ci. Pour cela, il est usuel de le maintenir dans les opérateurs relationnels primaires.

Ceci n'empêche pas pour autant l'abandon du principe plus général. Une syntaxe spécifique, semblable à celle fréquemment utilisée en mathématique pour le même usage (*soit* $x = \text{exp} : f(x)$), peut être définie. Cet opérateur est fréquemment présent dans les langages relationnels, par exemple, en SQL :

- **WITH x AS (exp) f(x)**

Note sur la projection

Le langage Tutorial D dénote la projection par la juxtaposition d'une (dénotation de) relation et d'une (dénotation de) liste d'attributs :

- $R \{a_1, \dots, a_n\} \equiv R \pi \{a_1, \dots, a_n\}$

Cet usage peut être vu comme analogue à la dénotation de la multiplication en arithmétique par juxtaposition des deux opérandes. Par souci de lisibilité, nous éviterons le plus souvent d'utiliser ce raccourci notionnel.

Note sur le produit cartésien

Dans ses premières publications, Codd utilisait un ensemble d'opérateurs relationnels primaires comprenant le produit cartésien plutôt que la jointure. Sachant que le produit cartésien est un cas particulier de la jointure et que la jointure peut être exprimée en combinant le produit cartésien et la restriction, les deux ensembles sont donc équivalents. Nous préférons la version présentée ici pour des raisons que nous exposerons bientôt.

Note sur l'algèbre minimale A

On peut réduire l'ensemble des opérateurs à un noyau minimal composé des seuls deux opérateurs suivants: $\langle \text{NAND} \rangle$ et $\langle \text{REMOVE} \rangle$. La démonstration en est faite par Date et Darwen dans le 3^e manifeste.

Plan de la section

Les prochaines sous-sections présentent la définition formelle des opérateurs relationnels primaires (3.1), puis la construction, à partir de ceux-ci, d'opérateurs usuels (3.2), spécialisés (3.3) et de regroupement (3.4).

Note de LL

Les prochaines sections présentent un aperçu de la mise en correspondance entre les langages proprement relationnels et SQL, le lecteur trouvera un traitement plus complet dans [5]. Une confusion certaine est encore manifeste dans la notation des langages relationnels. La notation de l'algèbre relationnelle est fluctuante selon les auteurs (ainsi Elmasri, Ullman, Date et Codd utilisent chacun des notations différentes). Il en est de même des langages relationnels, tels que Discipulus (langage en cours de définition au sein du groupe Μητις — ici composé en **STIX Two Text de couleur ocre**) et Tutorial D (langage défini par Date, mais toujours en évolution — ici composé en **Courier de couleur verte**). Quant à SQL (ici composé en **Droid Sans Mono de couleur pourpre**), nous nous référons à la syntaxe prescrite par la norme ISO/IEC 9075:2016 (qu'aucun SGBD ne propose sans variation dialectale plus ou moins importante).

3.1. Opérateurs primaires

Définition relativement à la théorie des ensembles et à la représentation précédente (au modèle, à la structure).

3.1.1. Renommage

$R \rho A:C$

A	B		C	B
a1	b1	$\rho A:C =$	a1	b1
a2	b2		a2	b2
a3	b3		a3	b3

Soit R une relation,
soit a et b deux identifiants alors

$R \rho a:b \equiv$
 $R \text{ RENAME } \{a \text{ AS } b\} \equiv$
 $\text{select } a_1, \dots, a \text{ as } b, \dots, a_n \text{ from } R \equiv$
 $\text{ANTE } a \in \text{id}(R) \wedge b \notin \text{id}(R):$
 $\text{SOIT } E := \text{def}(R, a); Z := \text{def}(R) - \{a:E\} \cup \{b:E\}:$
 $(Z; \{(Z; \text{val}(t) - \{(a, \text{val}(t, a))\} \cup \{(b, \text{val}(t, a))\}) \mid t \in \text{val}(R)\})$

Corolaires

- $\text{card}(R) = \text{card}(R \rho a:b)$
- $\forall t \in \text{val}(R) : (Z; \text{val}(t) - \{(a, \text{val}(t, a))\} \cup \{(b, \text{val}(t, a))\}) \in R \rho a:b$
- $a \notin \text{id}(R \rho a:b) \wedge b \in \text{id}(R \rho a:b)$
- $R = ((R \rho a:b) \rho b:a)$

Notes

- La présence de l'entête dans chacun des tuples et chacune des relations permet de définir une opération structurelle, le renommage (applicable tant aux tuples qu'aux relations).
- L'opération de renommage n'est bien définie que si l'identifiant à changer est défini dans l'entête et que le nouvel identifiant ne l'est pas.
- L'entête d'une relation est conservé dans le catalogue du SGBDR. Le catalogue est la description des modèles relationnels du SGBDR sous la forme d'une BD dont le modèle relationnel est lui-même dans le catalogue, comme tous les autres modèles relationnels de toutes les autres BD du SGBDR.

3.1.2. Projection

$R \pi \{A, C\}$

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3

Soit w une expression ayant pour valeur une liste d'identifiants d'attribut $\{w_0, \dots, w_n\}$ alors

$R \pi w \equiv$
 $R \{w_0, \dots, w_n\} \equiv$
 $\text{select } w_1, \dots, w_n \text{ from } R \equiv$
 $\text{ANTE } w \subseteq \text{id}(R):$
 $\text{SOIT } Z := \{a:D \mid a \in \text{id}(R) \wedge a \in w\}:$
 $(Z; \{(Z; \{(a, v) \mid (a, v) \in \text{val}(t) \wedge a \in w\}) \mid t \in \text{val}(R)\})$

L'opération de projection (applicable tant aux tuples qu'aux relations) n'est bien définie que si tous les identifiants d'attributs sont définis dans l'entête de la relation.

Note

- En Tutorial D et en SQL, la liste doit être explicite.

3.1.3. Jointure

$R \bowtie S$

A	B		B	C		A	B	C
a1	b1		b1	c1		a1	b1	c1
a2	b1		b2	c2		a2	b1	c1
a3	b3		b3	c3		a3	b3	c3
a4	b4		b3	c4		a3	b3	c4

Soit R et S deux relations alors

$R \bowtie S \equiv$

$R \text{ JOIN } S \equiv$

`select * from R natural join S` \equiv

$SOIT X := id(R) \cap id(S) :$

$ANTE \forall a \in X : a:D \in def(R) \wedge a:D \in def(S) :$

$SOIT Z := def(R) \cup def(S) :$

$(Z; \{(Z; val(t1) \cup val(t2)) \mid$

$t1 \in val(R) \wedge t2 \in val(S) \wedge (\forall a \in X : (a,v) \in val(t1) \wedge (a,v) \in val(t2))\}$

Note

- La jointure naturelle n'est pas commutative en SQL en raison de l'ordonnancement des attributs dans un tuple, puisque le tuple y est défini comme une liste ordonnée d'attributs et non comme un ensemble d'attributs.

3.1.4. Restriction

$R \sigma \text{ cond}$

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3

Soit R une relation,

soit c , une condition (expression booléenne),

soit $id(c)$, l'ensemble des identifiants d'attribut utilisés par c alors

$R \sigma c \equiv$

$R \text{ WHERE } c \equiv$

`select * from R where c` \equiv

$ANTE id(c) \subseteq id(R) :$

$(def(R); \{t \mid t \in val(R) \wedge c(t)\})$

L'opération de restriction n'est bien définie que si tous les identifiants d'attributs de la condition sont

définis dans l'entête de la relation.

3.1.5. Union

$R \cup S$

A	B
R	
S	

Soit R et S deux relations alors

$R \cup S \equiv$

$R \text{ UNION } S \equiv$

$\text{select } * \text{ from } R \text{ union select } * \text{ from } S \equiv$

$\text{ANTE } \text{def}(R) = \text{def}(S) :$

$(\text{def}(R); \text{val}(R) \cup \text{val}(S))$

Note

- L'antécédent doit être aménagé en cas de relaxation de la compatibilité.

3.1.6. Différence

$R - S$

A	B
R	
S	

Soit R et S deux relations alors

$R - S \equiv$

$R \text{ MINUS } S \equiv$

$\text{select } * \text{ from } R \text{ except select } * \text{ from } S \equiv$

$\text{ANTE } \text{def}(R) = \text{def}(S) :$

$(\text{def}(R); \text{val}(R) - \text{val}(S))$

Note

- L'antécédent doit être aménagé en cas de relaxation de la compatibilité.

3.1.7. Synthèse des opérateurs relationnels primaires

Restriction

$$R \sigma_{\text{cond}}$$

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3

Projection

$$R \pi_{\{A, C\}}$$

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3

Jointure

$$R \bowtie S$$

A	B
a1	b1
a2	b1
a3	b3
a4	b4

 \bowtie

B	C
b1	c1
b2	c2
b3	c3
b3	c4

 $=$

A	B	C
a1	b1	c1
a2	b1	c1
a3	b3	c3
a3	b3	c4

Différence

$$R - S$$

A	B
R	
S	

Union

$$R \cup S$$

A	B
R	
S	

Renommage

$$R \rho_{A:C}$$

A	B
a1	b1
a2	b2
a3	b3

 $\rho_{A:C} =$

C	B
a1	b1
a2	b2
a3	b3

3.2. Opérateurs usuels

Afin d'augmenter l'expressivité de l'algèbre relationnelle, il est possible de définir d'autres opérateurs en terme des opérateurs relationnels primaires. En voici quelques-uns.

3.2.1. Intersection

$$R \cap S = R - (R - S) = R \bowtie S$$

A	B
R	
S	

Soit R et S deux relations :

$R \cap S \equiv$

$R \text{ INTERSECT } S \equiv$

$\text{select } * \text{ from } R \text{ intersect select } * \text{ from } S \equiv$

ANTE $\text{def}(R) = \text{def}(S)$:

$R \bowtie S$

Notes

- L'antécédent garantit que $R \bowtie S = R - (R - S)$.
- L'antécédent doit être aménagé en cas de relaxation de la compatibilité.

3.2.2. Produit (cartésien)

$$R \times S = R \bowtie S$$

A		B		A	B
x		a		x	a
y		b		x	b
		c		x	c
				y	a
				y	b
				y	c

Soit R et S deux relations alors

$R \times S \equiv$

$R \text{ TIMES } S \equiv$

$\text{select } * \text{ from } R \text{ cross join } S \equiv$

ANTE $\text{id}(R) \cap \text{id}(S) = \emptyset$:

$R \bowtie S$

Notes

- L'antécédent garantit que $R \bowtie S = R \times S$.
- L'antécédent doit être aménagé en cas de relaxation de la compatibilité.
- Le produit cartésien n'est pas commutatif en SQL.

3.2.3. Semi-jointure (matching)

$$R \ltimes S = (R \bowtie S) \pi R$$

A	B		B	C		A	B
a1	b1		b1	c1		a1	b1
a2	b1		b2	c2		a2	b1
a3	b3		b3	c3		a3	b3
a4	b4		b3	c4			

Soit R et S deux relations alors

$R \bowtie S \equiv$

$R \text{ MATCHING } S \equiv$

$\text{select } r_1, \dots, r_n \text{ from } R \text{ natural join } S \equiv$

$(R \bowtie S) \pi_{id(R)}$

3.2.4. Semi-différence (not matching)

$$R \ltimes S = R - (R \bowtie S)$$

A	B		B	C		A	B
a1	b1	\ltimes	b1	c1	=	a4	b4
a2	b1		b2	c2			
a3	b3		b3	c3			
a4	b4		b3	c4			

Soit R et S deux relations alors

$R \ltimes S \equiv$

$R \text{ NOT MATCHING } S \equiv$

$\text{select } * \text{ from } R \text{ except select } r_1, \dots, r_n \text{ from } R \text{ natural join } S \equiv$

$R - (R \bowtie S)$

3.2.5. Extension

$$R \xi \{C := f(A, B)\} = (R \bowtie F) \rho_{f':C}$$

A	B		A	B	C
a1	b1	$\xi \{C := f(A, B)\} =$	a1	b1	f(a1, b1)
a2	b1		a2	b1	f(a2, b1)
a3	b3		a3	b3	f(a3, b3)
a4	b4		a4	b4	f(a4, b4)

Définition

Une relation F correspond à une fonction $f(p_1:T_1, \dots, p_n:T_n)$ de type T lorsque

$$\text{def}(F) = \{p_1:T_1, \dots, p_n:T_n, f:T\} \wedge F = (F \sigma_{f'=f(p_1, \dots, p_n)})$$

3.2.5.1. Solution utilisant les seuls opérateurs relationnels primaires

Soit R une relation,

soit F la relation correspondant à la fonction $f(p_1:T_1, \dots, p_n:T_n)$ de type T alors

$R \xi \{c := f(p_1, \dots, p_n)\} \equiv$

$R \text{ EXTEND } \{c := f(p_1, \dots, p_n)\} \equiv$

$\text{select } R.*, f(p_1, \dots, p_n) \text{ as } c \text{ from } R \equiv$

ANTE

$c \notin id(R) \wedge$

$\text{def}(R) \supseteq \text{def}(F) - \{f:T\} \wedge$

$R \pi_{\{p_1, \dots, p_n\}} \subseteq F \pi_{\{p_1, \dots, p_n\}} :$

$R \bowtie (F \rho_{f':c})$

Notes

- L'antécédent signifie simplement que
 - (1) l'identifiant c ne dénote pas déjà un attribut de R ,
 - (2) chaque paramètre de F correspond à un attribut de R ,
 - (3) tout tuple de R à son correspondant dans F .
- La construction de relations en lieu et place des fonctions est un élément important dans la démonstration de la complétude de l'algèbre relationnelle.
- Le symbole utilisé pour l'extension varie beaucoup d'un auteur à l'autre. Certains utilisent α . D'autres présentant l'opération comme une extension de la projection et utilisent conséquemment le symbole π .

Exercice

- En pratique, plusieurs affectations peuvent être spécifiées dans la portée de l'extension, elles sont réputées être réalisées indépendamment (c'est-à-dire que toutes les fonctions sont d'abord évaluées, puis les affectations réalisées en bloc) ; il est donc possible de les exécuter concurremment.
 - En conséquence, $R \xi \{r_1 := \text{exp}_1 ; r_2 := \text{exp}_2\}$ pourrait ne pas être équivalent à $(R \xi \{r_1 := \text{exp}_1\}) \xi \{r_2 := \text{exp}_2\}$ en particulier si exp_2 fait appel à l'attribut r_1 .
- Qu'en est-il en SQL ? Par exemple, quel est le résultat de

```
select 5 as x, x as y
from (values (1, 2)) as A (x, y) ;
```

3.2.5.2. Solution utilisant le modèle

Soit R une relation,

soit $f(p_1:T_1, \dots, p_n:T_n)$ une fonction de type T alors

$R \xi \{c := f(p_1, \dots, p_n)\} \equiv$

$R \text{ EXTEND } \{c := f(p_1, \dots, p_n)\} \equiv$

$\text{select } *, f(p_1, \dots, p_n) \text{ as } c \text{ from } R \equiv$

ANTE

$c \notin \text{id}(R) \wedge$

$\{p_1, \dots, p_n\} \subseteq \text{id}(R) :$

SOIT

$Z := \text{def}(R) \cup \{c:T\} :$

$(Z; \{(Z; \text{val}(t) \cup \{(c, f(t.p_1, \dots, t.p_n))\}) \mid t \in \text{val}(R)\})$

3.2.6. Image

Soit R une relation et t un tuple alors

$R \circ t \equiv$

$R(t) \equiv$

$\text{IMAGE_OF } t \text{ IN } R \equiv$

$R \bowtie \text{RELATION}[t \pi(\text{id}(t) \cap \text{id}(R))]$

Soit R une relation et t un tuple,

soit $\{x_1, \dots, x_n\} := \text{id}(t) \cap \text{id}(R)$ alors

$R \circ t \equiv$

$\text{select } * \text{ from } R \text{ where } R.x_1=t.x_1 \text{ and } \dots \text{ and } R.x_n=t.x_n$

Note

- Si R et t n'ont pas d'attributs communs ($n=0$), la condition est vraie!

3.2.7. Image du tuple courant

Définition

Au sein d'une expression relationnelle référant à une relation R , le tuple « courant » d'une relation R a pour valeur le tuple de la relation R effectivement utilisé au moment de l'évaluation de ladite expression. Il est noté $R[*]$ en Discipulus (ou $TUPLE\{*\}$ IN R en TutorialD).

L'image du tuple courant d'une relation R (donc la valeur de relation contenant ce seul tuple courant) est notée $!!R$ en Discipulus et en TutorialD. C'est-à-dire :

$$!!R \equiv R \circ R[*]$$

$$!!R \equiv IMAGE_OF\ TUPLE\{*\}\ IN\ R$$

En SQL, il faut recourir à une dénotation énumérative pour désigner le tuple courant

Soit $\{x_1, \dots, x_n\} := id(R)$ alors

$$R[*] \equiv (x_1, \dots, x_n)$$

$$!!R \equiv (values(x_1, \dots, x_n))$$

3.3. Opérateurs spécialisés

À développer

3.4. Opérateurs de regroupement

À développer

3.5. Opérateurs d'agrégation

Un opérateur d'agrégation (ou fonction d'agrégation) est traditionnellement présenté comme un opérateur dont les paramètres sont des « colonnes ». Bien que cette métaphore puisse guider l'intuition de façon satisfaisante dans les cas simples, elle ne rend pas compte de toutes les possibilités couvertes par ces opérateurs. De plus, elle fait appel à un concept, la « colonne » (qu'on aurait tort d'identifier à l'attribut) qui n'est pas définie dans le modèle relationnel. Il faut plutôt considérer l'opérateur d'agrégation comme une fonction définie sur une relation et *certain*s de ces attributs. En couplant de telles fonctions avec les autres opérateurs relationnels, il est possible de couvrir non seulement les fonctions d'agrégation, mais aussi les fonctions de fenêtrage (*window functions*) et la clause *lateral* de SQL.

Nous laissons pour le moment la définition formelle en exercice et renvoyons à [Date2015a] pour la solution.

3.6. Opérateurs de graphes

À développer

Les graphes sont en fait des relations binaires internes. En ce sens tous les opérateurs requis pour les manipuler sont déjà disponibles (et même plus) par l'entremise des opérateurs primaires. Il n'en demeure pas moins qu'il est possible, et souvent très commodes, de définir des opérateurs spécifiques tirant parti de l'internalité, de la binarité et de la transitivité cololaire.

3.7. Opérateurs déclassés

- division, DIVIDE BY...
- sommaire, SUMMARIZE...

Conclusion

Merci aux membres de CoLOED, CoFELI et Μητις :-))

Définitions

Sources consultées de juin 2023 à juillet 2024

- * Antidote: Antidote 11 v4.2 (2023), voir <https://www.antidote.info>
- * Le Larousse: <https://www.larousse.fr/dictionnaires/francais>
- * Le Robert: <https://dictionnaire.lerobert.com>
- * Wikipédia: <https://fr.wikipedia.org/wiki>

algèbre

Branche des mathématiques qui étudie les structures abstraites en employant les lois de composition.

informatique

1. Science
 1. Science du traitement de l'information.
 2. Science du traitement automatique et rationnel de l'information.
2. Technique
 - Ensemble des techniques de la collecte, du tri, de la mise en mémoire, du stockage, de la transmission, et de l'utilisation des informations traitées automatiquement à l'aide de logiciels mis en oeuvre sur des ordinateurs.
3. Spécialisation
 - Informatique théorique: concerne la définition de concepts et modèles.
 - Informatique pratique: s'intéresse aux techniques concrètes de mise en oeuvre.
 - Informatique appliquée: s'intéresse à l'utilisation de l'informatique pour la résolution de problèmes formulés en regard d'autres domaines (que l'informatique).

Corolaires

L'information ne peut être traitée automatiquement que si elle est représentée par des données, elles-mêmes réalisées sous la forme d'un phénomène tangible (physique).

Par ailleurs, l'information est notamment relative à l'espace-temps et à l'agent.

D'autre part, le traitement est décrit par des procédés et rendu effectif par des processus dont les actions atomiques doivent être dénombrables (ne serait-ce que porter le critère d'efficience). Si, en pratique, il est utile de pouvoir leur associer une durée, la mesure du temps n'est pas pour autant requise, bien que fréquente.

En conséquence,



- le coeur de métier détermine la modélisation (de l'information et du traitement); dans le cas du traitement, il est suffisant d'étudier (de modéliser) les seuls procédés de traitement de l'information, par contre, par définition, la modélisation de l'information doit s'appliquer à tout type d'information, donc à toute information relative à l'univers physique;
- (toutes) les grandeurs physiques doivent donc être prises en compte (même si pour l'information non connotée physiquement, il pourrait être jugé suffisant de se limiter à l'espace et au temps — c'est le cas en informatique théorique);
- une grandeur propre à l'informatique doit être ajoutée aux grandeurs physiques afin de mesurer l'information: le bit (une autre unité sera peut-être requise pour l'information quantique).

Débats



- Le bit mesure-t-il l'information, la donnée, une représentation spécifique de la donnée?
- Les unités mesures (et leur définition) du temps et de l'espace utilisée en informatique doivent-elles être les mêmes qu'en physique?
- Plusieurs priorités du temps sont encore très débattues, en physique comme en informatique:
 - Le temps est-il discret ou continu?
 - Est-il borné (dans le passé, dans le futur)?
 - Est-il cyclique?
 - Le moment de référence du 1^{er} janvier 1970 souvent retenu en informatique et en physique, doit-il être à midi ou à minuit?
 - Ce moment de référence ne devrait-il pas être le 1^{er} janvier 1958? Pourquoi?
 - Le temps doit-il être coordonné ou non?
 - Faut-il adopter une notation qui s'affranchit complètement des calendriers, comme la mesure de la longueur s'est affranchie de la morphologie humaine (pouce, paume, pied) et celle de la masse, d'un objet-étalon (le mètre-étalon)?

information

Élément de connaissance susceptible d'être transmis au moyen d'une suite de signes.

Élément de connaissance représentable par une donnée.

logique

- Antidote: Science qui a pour objet l'étude des méthodes de raisonnement, de pensée, par lesquelles on peut atteindre la vérité.
- Larousse: Science du raisonnement en lui-même, abstraction faite de la matière à laquelle il s'applique et de tout processus psychologique.
- Le Robert: Étude scientifique, surtout formelle, des normes de la vérité.
- Wikipedia: Étude des règles formelles que doit respecter toute argumentation correcte.

représentation

Suite de signaux.

type

Définition variable selon les auteurs.

LL

Selon la théorie des types (Russell):

- defA: dénotation d'un ensemble de valeurs propres (selon le modèle, l'ensemble peut être infini, ou pas).
- defB: dénotation d'un sous-ensemble de defA déterminé par une contrainte (selon le modèle, le sous-ensemble peut être impropre, ou pas).



Suivant Russell, la plupart des logiciens, des mathématiciens et des informaticiens algorithmiques ont adopté les dénominations suivantes:

- Type: defA
- Sous-type: defB

Toutefois, la plupart des ontologistes et des informaticiens en modélisation de données (dont Codd et Date première manière) utilisent les dénominations suivantes:

- Domaine : defA
- Type : defB

En SQL, on augmente encore la confusion :

- Type : defA
- Domaine : defB

À noter que Date s'est rallié à la dénomination de Russell dans «Type Inheritance and the Relational Theory» (2016).

Dans les documents du CoFELI, nous utiliserons la dénomination de Russell et consorts.

Sigles

ACID

Acronyme désignant conjointement les propriétés d'atomicité, de cohérence, d'isolation et rémanence (ou *durability* en anglais).

EA

Acronyme désignant les modèles conceptuels de données fondés sur la théorie entité-association.

SGBD (Système de gestion de bases de données)

Service informatique permettant...

SGBDR (Système de gestion de bases de données relationnelles)

Service informatique permettant...

SQL (*Structure Query Language*)

Langage de programmation axiomatique fondé sur un modèle inspiré de la théorie relationnelle proposée par E. F. Codd.

[Normes applicables : ISO 9075:2016, ISO 9075:2023]

Références

[Codd1970a]

Edgard F. CODD;
A Relational Model of Data for Large Shared Data Banks;
Communications of the ACM, 13(6), pp. 377–387, 1970;
doi:10.1145/362384.362685.

[Codd1990a]

Edgard F. CODD;
The Relational Model for Database Management: Version 2;
Addison-Wesley Longman Publishing, Boston (MA, USA), 1990;
ISBN 0-201-14192-2.

[Date1998a]

Chris J. DATE, Hugh DARWEN;
First Edition : *Foundation for Object/Relational Database Systems: The Third Manifesto*;
Addison-Wesley Redwood City (CA, US), 1998; ISBN 0-201-30978-5.
Second Edition : *Foundation for Future Database Systems: The Third Manifesto*;
Addison-Wesley, Redwood City (CA, US), 2000; ISBN 0-201-70928-7.
Third edition : *Databases, types, and the relational model: The third manifesto*;
Addison-Wesley (Pearson Education), 2007; ISBN 0-321-39942-0.
Third revised edition : *Databases, types, and the relational model: The third manifesto*;
2014, <https://www.dcs.warwick.ac.uk/~hugh/TTM/DTATRM.pdf> (consulté le 2024-05-30).

[Date2015a]

Chris J. DATE;
SQL and relational theory: how to write accurate SQL code;
O'Reilly Media, 2015;
ISBN 978-1-4919-4117-1.

[Date2020a]

Chris J. DATE;
Logic and Relational Theory;
Technics Publications, Basking Ridge (NJ, US), 2020;
ISBN 978-1634628754.

[Russell1908a]

Bertrand RUSSELL;
Mathematical Logic as Based on the Theory of Types;
American Journal of Mathematics, vol. 30, no 3, 1908, p. 222–262;
ISSN 0002-9327, DOI 10.2307/2369948.

Produit le 2025-01-09 09:53:56 UTC

