

## Breves notas Sobre la Semántica de las Dependencias de Datos

Se define una clase unificada de dependencias: las que generan igualdades y las que generan tuplas; y se unifica la semántica de estas dependencias.

### Compendio Introductorio a las Dependencias de los Datos

La especificación de constricciones que han de satisfacer los datos es un importante asunto del diseño de esquemas de las bases de datos relacionales que modelan partes del mundo real. Estas constricciones son las que determinan si el significado de una base de datos es válido o no.

Las constricciones llamadas "dependencias de datos", o "dependencias" de forma abreviada, son de particular interés. El estudio de las dependencias empezó con la *dependencia funcional* en [8]. Después de la introducción de las *dependencias multivaluadas* en [9] y [25], el campo se hizo caótico. En poco tiempo se introdujeron muchas nuevas clases de dependencias, por ejemplo, *dependencias mutuas* [14], *dependencias de join* [2,19], *dependencias transitivas* [16], *dependencias generales* [18], y *dependencias de subconjuntos* [21].

Sin embargo, todas estas dependencias tienen una semántica similar. Intuitivamente, el significado de una dependencia es que si existen en la BD algunas tuplas que satisfacen ciertas igualdades, entonces algunas otras tuplas tienen forzosamente que existir también en la BD, o algunos valores en las tuplas dadas tienen que ser iguales. Este compendio introduce la clase de *dependencias-de-generación-de-tuplas y de-generación-de-igualdad*, que van a capturar esta semántica intuitiva. Esta clase incluye la mayoría de las clases de interés. La idea es más de unificación que de generalización. Y permite tratar con dos clases de dependencias, en vez de con un gran número de clases de dependencias diferentes.

Casi todos los trabajos teóricos sobre dependencias tratan exclusivamente con distintos aspectos del *problema de la implicación*, es decir, el problema de decidir cuándo un conjunto dado de dependencias implica lógicamente otra dependencia. La razón de la relevancia de este problema es que un algoritmo para decidir la implicación de dependencias nos permite decidir cuándo son equivalentes dos conjuntos dados de dependencias o cuándo es redundante un conjunto dado de dependencias. Una solución para los dos últimos problemas es un paso significativo hacia la obtención de un diseño automatizado de esquemas de BDs [3,4,7, 12,23], que se percibe como el objetivo final de la investigación en la teoría de dependencias [2].

En [1] y [13] se desarrolla un procedimiento de decisión<sup>1</sup> llamado *chase* para el problema de la implicación en las dependencias funcional y de *join*. Este procedimiento se generaliza en [6] para las *dependencias-de-generación-de-tuplas y de-generación-de-igualdad*, y presenta un procedimiento de demostración del problema de la implicación para estas dependencias<sup>2</sup>. En algunos casos, sin embargo, *chase* es también un procedimiento de decisión; por ejemplo, si todas las dependencias son totales. En algunos casos se puede ver cómo *chase* puede llevar a un procedimiento de decisión eficiente [6].

Las *dependencias-de-generación-de-tuplas y de-generación-de-igualdad* se pueden expresar como sentencias de primer-orden [10, 15, 22]. Así, se puede argumentar que no es necesario desarrollar un procedimiento de demostración para las dependencias, ya que cualquier procedimiento de demostración para la lógica de primer-orden lo hará. Sin embargo, las dependencias son exactamente un fragmento de la lógica de primer orden, un fragmento que parece ser adecuado para expresar las reglas de integridad de las bases de datos, y deseamos tener un procedimiento de demostración que esté especializado para este fragmento. Este procedimiento de demostración especializado es bastante útil. Por ejemplo, en [6] se utiliza para demostrar varios resultados de "decidibilidad", y en [5] se usa para inventar un sistema formal para las dependencias.

---

<sup>1</sup> Se distingue entre un *procedimiento de decisión* que siempre acaba y un *procedimiento de demostración* que puede estar en ejecución indefinida cuando se tiene respuesta negativa al problema de decisión.

<sup>2</sup> Generalizaciones similares se estudian en [11,17,20 y 24]

## 2. REFERENCIAS BIBLIOGRÁFICAS

1. AHO, A. V., BEERI, C., AND ULLMAN, J.D. The theory of joins in relational databases. *ACM Transactions on Database Systems* 4, 3 (Sept. 1979), 297-314.
2. BEERI, C., BERNSTEIN, P. A., AND GOODMAN, N. A sophisticate's introduction to database normalization theory. In *Proc. of the 4th Int. Conf. on Very Large Data Bases* (Berlin). ACM, New York, 1978, pp. 113-124.
3. BEERI, C., MENDELZON, A. O., SAGIV, Y., AND ULLMAN, J.D. Equivalence of relational database schemes. *SIAM J Comput* 10(1981), 647-656.
4. BEERI, C., AND RISSANEN, J. Faithful representation of relational database schemes. IBM Res. Rep. IBM, San Jose, 1980.
5. BEERI, C., AND VARDI, M.Y. Formal system for tuple and equality generating dependencies. *SIAM J Comput* 13 (1984), 76-98.
6. BEERI, C., AND VARDI, M.Y. A Proof Procedure for Data Dependencies, J. ACM Vol 31, No. 4, Oct. 1984, p 718-741.
7. BERSTEIN, P.A. Synthesizing third normal form relations from functional dependencies. *ACM Trans Database Systems* 1, 4 (Dec. 1976), 277-298.
8. CODD, E.F. Further normalization of the data base relational model. In *Data Base Systems* (R. Rustin, Ed.). Prentice-Hall, Englewood Cliffs, N.J., 1972, pp. 33-64.
9. FAGIN, R. *Multivalued dependencies and a new normal form for relational databases*. *ACM Trans. Database Syst* 2, 2 (June 1977), 262-278.
10. FAGIN, R. Horn clauses and database dependencies. *J. ACM* 29, 4 (Oct. 1982), 952-985.
11. GRANT, J., and JACOBS, B.E. On the family of generalized dependency constraints. *J. ACM* 29, 2 (Oct. 1982), 986-997.
12. MAIER, D., MENDELZON, A. O., SADRI, F., AND ULLMAN, J.D. Adequacy of decompositions of relational databases. In *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds. Plenum Press, New York, 1981, pp. 101-114.
13. MAIER, D., MENDELZON, A. O., AND SAGIV, Y. Testing implications of data dependencies. *ACM Trans. Database Systems*. 4, 4 (Dec. 1979), 455-469.
14. NICOLAS, J.M. First order logic formalization for functional, multivalued and mutual dependencies. In *Proc. of the ACM-SIGMOD Int. Conf. on Management of Data* (Austin). ACM, New York, 1978, pp. 40-46.
15. NICOLAS, J.M. Mutual dependencies and some results on undecomposable relations. In *Proc. of the 4th Int. Conf. on Very Large Data Bases* (Berlin). ACM, New York, 1978, pp. 360-367.
16. PAREDAENS, J. *Transitive dependencies in a database scheme*. *RAIRO Inf. Comput Sci* 14(1980), 149-164.
17. PAREDAENS, J. A universal formalism to express decomposition, functional dependencies and other constraints in a relational database. *Theor. Comput. Sci.* 19 (1982), 143-160.
18. PAREDAENS, J., AND JANSSENS, D. Decompositions of relations---A comprehensive approach. In *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds. Plenum Press, New York, 1981, pp. 73-100.
19. RISSANEN, J. Theory of relations for databases--A tutorial survey. In *Proceedings of the 7th Symposium on Mathematical Foundations of Computer Science* (Zakopane, Poland) LNCS, vol. 64. Springer-Verlag, New York, 1978, pp. 537-551.
20. SADRI, F., AND ULLMAN, J.D. Template dependencies: A large class of dependencies in relational databases and its complete axiomatization. *J. ACM* 29, 2 (Apr. 1982), 363-372.
21. SAGIV, Y., AND WALECKA, S.F. Subset dependencies and a completeness result for a subclass of embedded multivalued dependencies. *J. ACM* 29, 1 (Jan. 1982), 103-117.
22. VARDI, M.Y. The implication problem for data dependencies in relational databases. Ph.D. Thesis (in Hebrew), The Hebrew University of Jerusalem, Jerusalem, Israel, 1981.
23. VARDI, M.Y. On the decomposition of relational databases. In *Proceedings of the 23rd, IEEE Symposium on the Foundations of Computer Science*, (Chicago, Nov. 3-5). IEEE, New York, 1982, pp. 176-187.
24. YANNAKAKIS, M., and PAPANASTASIOU, C. Algebraic dependencies. *J. Comput. System Sci.* 21(1982), 2--41.
25. ZANIOLO, C. Analysis and design of relational schemata for database systems. Tech. Rep. UCLA-ENG-7769. Dept. Computer Science, UCLA, Los Angeles, July 1976.