The Branching Data Model as a New Level of Abstraction over the Relational Database [†]

H. Paul Zellweger

ArborWay Labs, Rochester, MN 55901, USA; pz@arborwaylabs.com

[†] Conference Theoretical Information Studies (TIS), Berkeley, CA, USA, 2–6 June 2019.

Published: 19 May 2020

Abstract: Information is often stored in the relational database. This technology is now fifty years old, but there remain patterns of relational data that have not yet been studied. The abstract presents a new data pattern called the branching data model (BDM). It represents the pure alignment of the table's schema, its data content, and the operations on these two structures. Using a well-defined SELECT statement, an input data condition and its output values form a primitive tree structure. While this relationship is formed outside of the query, the abstract shows how we can view it as a tree structure within the table. Using algorithms, including AI, it goes on to show how this data model connects with others, within the table and between them, to form a new, uniform level of abstraction over the data throughout the database.

Keywords: branching data model; relational database; end-user navigation; named set theory

1. Introduction

The relational database was introduced by Codd fifty years ago [1]. Since that time, the research community has rigorously investigated many aspects of these systems. This study includes the data model [2], the design of the database schema, the ANSI standardization of its operations, and so on. However, uniform patterns in its raw data remain hidden, until now. The abstract outlines an arrangement of data relations in the relational database called the branching data model (BDM) initially presented in [3]. The organization of data in this data model represents a one-to-many data relationship that branches. It is a primitive tree structure. It has been shown that algorithms can aggregate this small structure to create larger trees automatically [4]. In turn, the raw data in this tree structure provide the display data for an end-user interface that can pinpoint sources of information in the database table based on the primary key [5], and database applications generated automatically [6].

The branching data model is motivated by named set theory [6]. Mark Burgin developed this new branch of theoretical mathematics to advance our understanding of the structure of a name and the concept of a set. The “algorithmic named set”, in particular, highlights the algorithmic flow between an input set and its output [7]. In notation, it is expressed by $\{I, X, O\}$, where I is an input set, O is an output set, and X depicts an algorithm between these two sets. In the database, table attributes and their domain values represent these source and destination sets. With a well-defined SQL SELECT statement, we can replace the input and output sets with the actual attribute labels from the table. In this context, the SELECT statement now serves as a level of abstraction over the underlying algorithms that perform the system's retrieval functions.

2. Materials and Methods

Consider a relational table Widgets that has multiple attributes. Any pair of attributes, such as COLOR and SIZE, can depict the input and output data states. Moreover, a SELECT statement, such as the one below, can represent the underlying algorithm that connects these two sets,

SELECT SIZE From Widgets WHERE COLOR = 'red'.

In our depiction of this query in a database table, the schema and its data values represent a static state. The Widgets table in Figure 1 below depicts this state as a snapshot. Our well-defined SELECT statement from above now identifies the input/output data points of the branching data model in this state.

COLOR	SIZE	WEIGHT	ID
red	small	light	001
red	medium	heavy	002
white	medium	heavy	003
blue	medium	light	004

Figure 1. The Widgets table.

First, the input condition COLOR = 'red' establishes a dataset in the table. Next, the output attribute SIZE returns data values that further divide this dataset into its subsets. Outside the SELECT query, program logic assigns the input data condition 'red' as a source and the output data 'small' and 'medium' a destination. See Figure 1 above for the graphic representation of this mapping between two attributes. In contrast, this arrangement of data in files and computer memory establishes the branching data model. Logically speaking, the BDM creates an IF-THEN deduction where the first element is a source, and its destination follows.

We can now model this pattern of data relations using its schema labels, "(COLOR, SIZE)". Once again, we use program logic outside of the query to establish a consistent, one-to-many pattern between these two attributes. In Figure 2 below, the graphics present this uniform pattern that maps COLOR data to its SIZE output.

COLOR	SIZE	WEIGHT	ID
red	small	light	001
red	medium	heavy	002
white	medium	heavy	003
blue	medium	light	004

Figure 2. The uniform pattern of the BDM between two attributes.

Named set theory predicts further expansion, whereby "(COLOR, SIZE)" overlaps with another BDM, such as "(SIZE, WEIGHT)" to model a data network in the table. With this expansion, the BDM now represents a well-defined chain of links that depict a data network in the database table. See Figure 3 below for the details of the expression "(COLOR, SIZE) (SIZE, WEIGHT) (WEIGHT, ID)". The data flow created by this BDM chain always connects to a single primary key. In a database interface, such as a list of nested list menus, the organization of data creates a "data funnel", whose menu paths pinpoints each unit of information in the table.

Figure 3. The BDM chain establishes a “data funnel”.

3. Using an Early Form of AI

And finally, a “brute-force” search, an early form of AI, replicates the BDM pattern between arbitrary pairs of tables in the database system. The algorithm does this by aligning pairs of primary and foreign keys between a source table and its destination, regardless of their intended relationship. In some instances, two tables can connect, and the modeling is stored in a directory to connect to table networks. In other cases, they do not. To date, we have yet to identify the “rules” on why these two tables relate, but, we believe, it depends upon the normalization of the database.

4. Conclusions

The extended abstract presents the branching data model (BDM) that establishes a new, uniform level of abstraction over the relational data in the database. This data model automates the construction of tree structures in the database. In the examples provided, these structures transform raw data into menu data for a database interface, known as a “data funnel.” End-users navigate its menu paths to locate sources of information in the database table. A well-defined SELECT statement establishes the BDM between the two table attributes. Outside of the query, program logic arranges a single data condition that self-references an attribute domain with its output values. In the BDM chain expression, each link in the progression goes from an input dataset to its output that divides this dataset into subsets. The final output is a primary key, a source of information in the database table. The advantage of this approach is twofold: (1) it represents a new uniform, abstract level of abstraction over relational data, and (2) it is a database retrieval device that models multiple SELECT statements automatically.

5. Patents

The most recent U.S. Patent allowed is 9,665,637 on 30 May 2017. More patents are pending.

Funding: This research received no external funding.

Acknowledgments: The author thanks Mark Burgin for all his overall support with this project.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Codd, E.F. A Relational Model of Data for Large Shared Data Banks. *Commun. ACM* **1970**, *13*, 377–387.
2. Codd, E.F. Data models in database management. *ACM SIGMOD Rec.* **1981**, *11*, 112–114.
3. Zellweger, H.P. The Branching Data Model, the Foundation for Automated Tree Visualization. In Proceedings of the 22nd International Conference on Information Visualization (IV2018), Salerno, Italy, 10–13 July 2018; pp. 33–37.
4. Zellweger, H.P. A Decision Tree Interface Based on Predicate Calculus. In Proceedings of the 21st International Conference on Information Visualization (IV’17), London, UK, 11–14 July 2017; pp. 188–193.
5. Zellweger, H.P. Tree Visualizations in Structured Data Recursively Defined by the Aleph Data Relation. In Proceedings of the 20th International Conference on Information Visualization (IV’16), Lisbon, Portugal, 19–22 July 2016; pp. 21–26.

6. Zellweger, H.P. A Demonstration of Automated Database Application Development. In Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI'19), Marina del Rey, CA, USA, 17–20 March 2019; pp. 65–66.
7. Burgin, M. *Theory of Named Sets*; Nova Science Publishing: New York, NY, USA, 2011.
8. Burgin, M. *Theory of Information; Algorithmic Named Sets*; World Scientific: Singapore, 2010; p. 568.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).