

# Architectural Support for Database Visualization

Dennis P. Groth  
Edward L. Robertson

Computer Science, Indiana University  
Bloomington, IN 47405, USA  
Email: {dgroth, edrbtsn}@cs.indiana.edu

## Introduction

The rapid proliferation and growth of database management systems has resulted in the retention of massive amounts of information for data processing and analysis needs. Many data processing requirements can be satisfied through the use of traditional database languages, such as SQL. These languages retrieve and present query results in record-oriented tables. The table of records format is best for presenting every record, but it cannot give a feel for the overall character of the data set.

This problem is similar to one faced in scientific computing - for example, in fluid dynamics. In that discipline, a computer simulation would produce literally millions of numbers; comprehension at this level was impossible. The development of high resolution graphics displays and fast graphics processors and programs allowing a scientist to navigate through visualizations of this data was required for fluid flow simulations to be truly useful.

We aim to achieve data visualization comparable to scientific visualization. However, scientific visualization has one major characteristic that data visualization lacks - scientific data has inherent dimensions, which connect to three spatial and one temporal dimension. Thus, it takes only elemental linear algebra to lay out scientific data on a two dimensional display. The challenge for data visualization is to achieve a comparably effective mapping from data to display. Keim [4,5,6] has achieved this effect with data from a single attribute. Shneiderman [1,2] has shown that significant productivity gains can be achieved by combining visualization techniques and database queries.

Database visualization differs from other types of information visualization due to the arbitrary nature of the data stored in the database. The attributes used to organize the presentation may not have a regular scale and, in fact, may not even have a semantically meaningful order. Data can be categorized based on whether it has inherent order or scale. For example, numeric data has both an order and a scale. In contrast, geographic data (latitude, longitude) can be easily scaled according to a well defined metric, yet ordering is not as straightforward. Figure 1 depicts the dichotomy between the ordering and scaling of data.

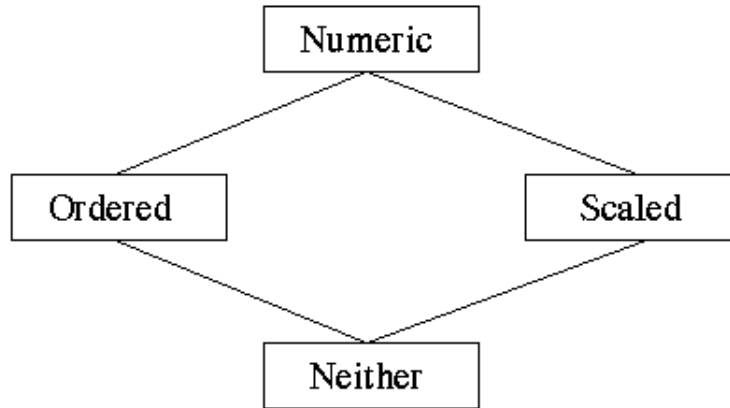


Figure 1: Contrasting order with metric scale

### Architecture

We posit an architecture to address the problem of database visualization. The essential and unique component of this architecture is the mapping functionality, which adds order and/or scale to the data. In addition, the architecture contains the following more conventional components: query specification, database storage, filtering, plotting and image display. Figure 2 shows a conceptual layout of the architecture.

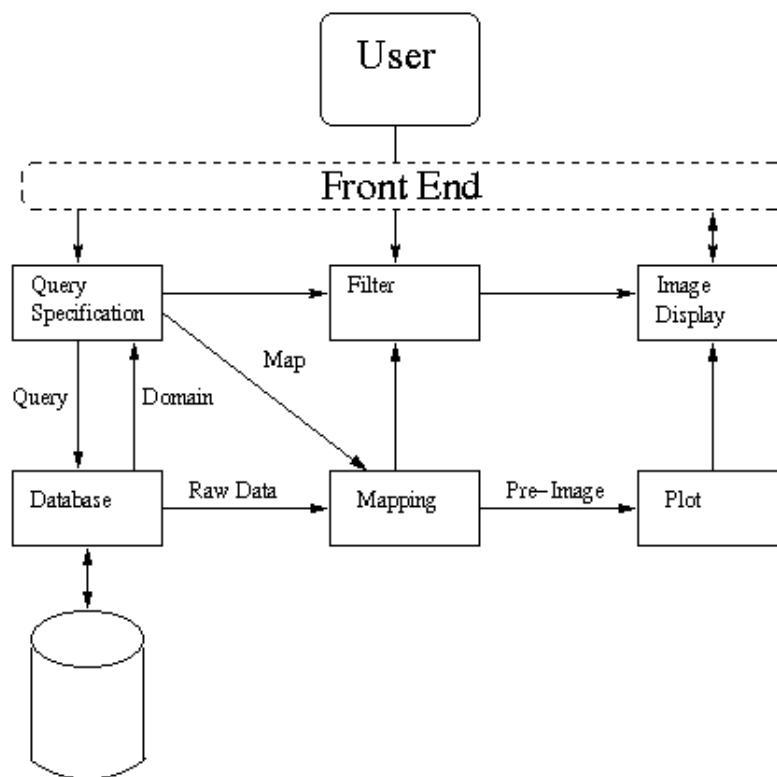


Figure 2: Database Visualization Architecture

Mapping is the key component of the architecture. In the context of our architecture, mapping is a function of the results generated by a query and a predefined **map**. A map maps domain values into a fixed numeric interval, thus providing both order and scale. A map may take into consideration the domain of the data as well as other information. The features of a particular map can be programmed to support a wide variety of visualizations. Naturally, the architecture supports the addition of new maps in a modular fashion. The map does not actually generate the screen image, however; that functionality is provided by the plot component, discussed below.

In Figure 3, the left image compares employee age (horizontal axis) and salary (vertical axis). While the age map is just the identity, the salary map maintains the natural numeric order but scales values using successive occurrences in the active domain. For example, if the active domain includes \$50,000, \$50,001, and \$60,000 in positions 100, 101, and 102 in the order, then these salaries are mapped to 100, 101 and 102; the difference in successive mapped positions is uniformly 1, in spite of huge discrepancies in the differences of the actual numeric values. Thus, the image preserves grouping in a compressed manner.

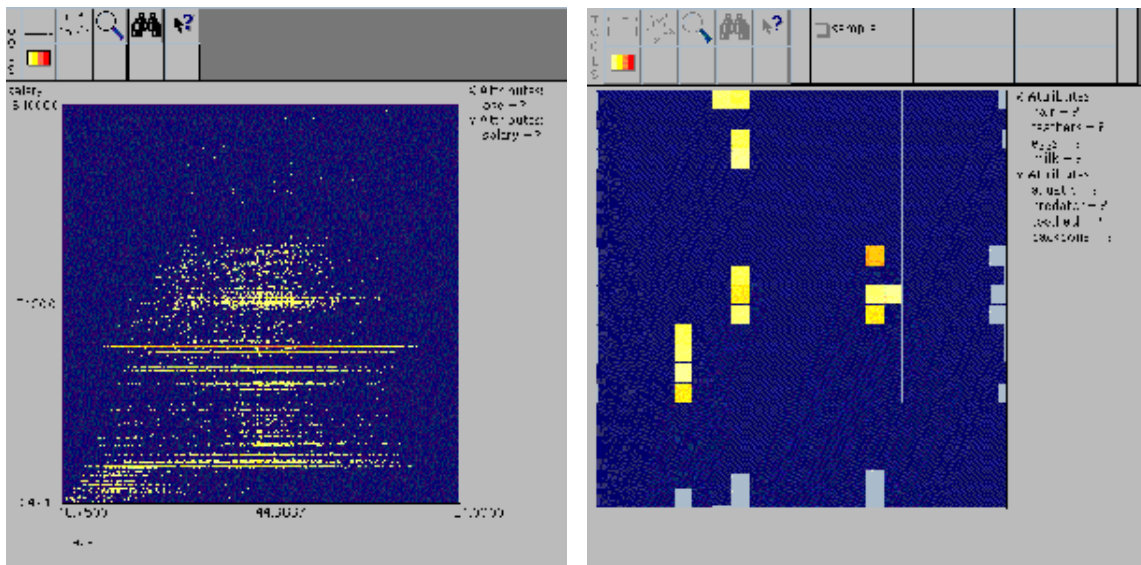


Figure 3: An numeric mapping compared to a boolean mapping

The right image illustrates that the map may be extended to handle several dimensions and can provide an unconventional order - in this case, Boolean values concatenated and then ordered according to a Gray code scheme. Using this map in both dimensions provides a Karnaugh map [3] layout and thus, a way to search for patterns of commonality when teamed with an interface that facilitates pivoting of the Boolean dimensions.

While we often map a particular value to a planar coordinate, we can also associate certain values to a range of colors. In Figure 3 we have indexed the population of values at a specific point to an RGB value. For example, light yellow points have few records, with higher populations represented by increasing intensities of red.

The plotting component of the architecture transforms the map coordinates into real coordinates based on the available screen "real estate." For example, in the right image of Figure 3, discrete points are

plotted as perceivable squares. In addition, the user may choose to apply filters to the visualization. A front end has been developed for interaction between the various components of the architecture and the end user. We recognize that the visualization process can be iterative, in that the user may view a particular query result and then refine their query or manipulate the visualization.

## Conclusion

In this paper we have introduced an architecture to support database visualization. Central to the architecture is the mapping component, which computes the data coordinates for the visualization based on a defined map. It is the visibility and manipulability of the map that distinguishes this work from other database visualization efforts. Future work will include the development of maps, optimization of the entire process, and usability evaluation.

## References

- [1] AHLBERG, C., SHNEIDERMAN, B. Dynamic queries for information exploration: an implementation and evaluation. In *Proceedings of the ACM CHI'92 Conference* (1992), pp. 619-626
- [2] AHLBERG, C., SHNEIDERMAN, B. Visual Information Seeking: Tight coupling of dynamic query filters with starfields display. In *Proceedings of the ACM CHI'94 Conference* (1994), pp. 313-317
- [3] KARNAUGH, M. The Map Method for Synthesis of Combinatorial Circuits *Transactions AIEE* 72 (1953) 593-598
- [4] KEIM, D. A. Visual Support for Query Specification and Data Mining. PhD Thesis, Institute for Computer Science, Ludwig-Maximilians-University Munich, 1995
- [5] KEIM, D. A., ANKERST, M., KRIEGEL, H. Visdb: Database Exploration Using Multidimensional Visualization. *IEEE Computer Graphics and Applications* 14, 5 (September 1994), 40-49
- [6] KEIM, D. A., KRIEGEL, H. Recursive Pattern: A Technique for Visualizing Very Large Amounts of Data. In *Proceedings of IEEE Visualization '95* (Los Alamitos, CA, October 1995), pp. 279-286