Scientific Computing Doesn't Need noSQL

David M. Butler
Limit Point Systems, Inc.
Livermore, CA 94550
d.m.butler@limitpoint.com

***Abstract*— We argue that the cluster of new data management technologies loosely known as "noSQL" are too low level to provide much value to scientific computing applications.**

***Keywords-** database management, data models, noSQL, scientific computing, high performance computing*

I. THE DATA MODEL PARADIGM

We use the term "data model" in the sense originally defined by Codd [1], that is, a data model consists of a class of mathematical objects, the operations on the objects, and a set of constraints on valid instances of the objects. Given a data model, we build languages, libraries, and tools based on the model. Applications are then developed in this higher level environment.

This development paradigm can produce numerous benefits. It increases the level of abstraction for application development. It increases the capability of applications. It facilitates interoperation and integration between diverse data sources. All of these benefits combine to increase the productivity of application programmers.

However, these benefits only accrue if the model actually captures some useful portion of the mathematical structure inherent in the applications. Furthermore, the more structure the data model captures, the greater the benefit. Hence, it is important that the data model capture as much of the application structure as possible.

II. SPECTRUM OF MATHEMATICAL STRUCTURE

With the preceding points in mind, it is useful to look at the spectrum of mathematical structure captured by various data models, as shown in Fig. 1. The figure shows various data models, the associated mathematical structure, and typical applications which use each model. Mathematical structure increases as one moves to the right in the figure.

Although the ranking of mathematical structure is only qualitative, several features are clear from the figure. First, most of the data models in the noSQL category have less mathematical structure than the relational model. Second, it is clear that scientific computing applications have much more mathematical structure than is captured by any of these models. It is already well known that the relational model isn't structured enough for scientific applications, the table operations are too low level, and so models with even less structure are unlikely to provide additional benefits. Thus the claim of the title: scientific computing doesn't need a no Structured Query Language

III. WHAT SCIENTIFIC COMPUTING NEEDS

Pretty clearly, what scientific computing needs is a much more Structured Query Language, which, to continue the naming theme, we can refer to as mo'SQL. The requirements mo'SQL must meet are the following.

The data model must capture the common mathematical structure of scientific applications. As suggested by the figure, there is a collection of mathematical abstractions commonly shared by scientific applications, especially in the physical sciences and engineering. These abstractions, which are inherited from mathematical physics, determine what the data means and how we want to use it. They include:

- physical property types (scalars, vectors, tensors)
- spatial types (topology and geometry)
- field types (property as a function of spatial position)
- algebra and calculus operations associated with each of the above

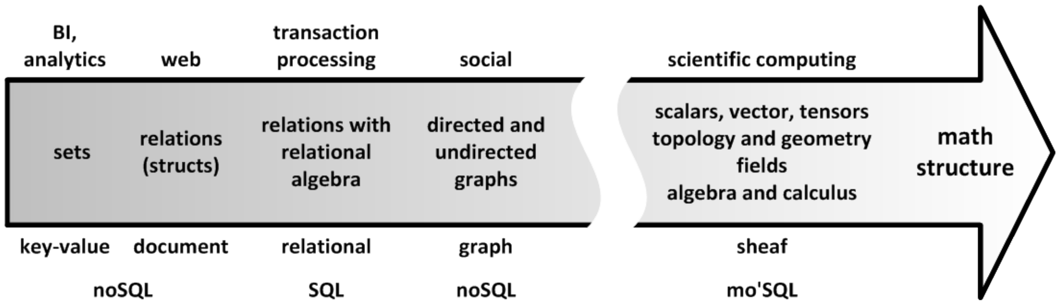

Figure 1. Spectrum of mathematical structure captured from application by data model.

The data itself is determined by how we represent these abstract objects. Hence, the data model must describe how we decompose the high level entities into collections of primitive types representable on the computer. In particular, the model must describe decomposition associated with parallelism.

Finally, the data model must maintain a rigorous connection between the data and the semantics of the abstract objects that the data represents.

IV. CONCLUSION

We can summarize all this very succinctly: scientific computing doesn't need noSQL, it needs mo'SQL.

REFERENCES

[1] E. F. Codd. 1970. A relational model of data for large shared data banks. Commun. ACM 13, 6 (June 1970), 377-387. DOI=10.1145/362384.362685 http://doi.acm.org/10.1145/362384.362685