Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 08-042

Infobionics Server - the next generation database

Prasanna Desikan, David Haggerty, Carl Bonta, and Jaideep Srivastava

December 05, 2008

Infobionics Server – the next generation database

Prasanna Desikan[1], David Haggerty[1], Carl Bonta[1], Jaideep Srivastava[2]

[1]Infobionics Inc, 7700 Equitable Drive, Suite #102, Eden Prairie, MN 55344

[2]Department of Computer Science, University of Minnesota, MN 55455

[1]{PrasannaDesikan, DavidHaggerty, CarlBonta}@infobionics.com

[2]srivasta@cs.umn.edu

ABSTRACT

This paper describes the 'Infobionics Server' - a next generation database. Also referred to as the 'Cellular Database Server', that is based on a novel 'cellular' data model.

Categories and Subject Descriptors

H.2.1 [**Database Management**]: Data models.

Keywords

Database, Data model, Flexible schema, Flexible query, Link.

1. INTRODUCTION

This paper presents a new data model and system that addresses the short comings and challenges faced in relational databases.

1.1 Motivation

Relational Model: In the very first paper on the relational model, Dr. Codd [1] described the term relation as:

"*Given sets S_1, S_2, ..., S_n (not necessarily distinct), R is a relation on those n sets if it is a set of n-tuples each of which has its first element from S_1, its second element from S_2, and so on. We shall refer to S_j as the j^{th} domain of R .As defined above, R is said to have degree n*".

This definition of relation specifies the way in which data is structured in the relational model – each row/tuple needs to have the same number of attributes and in the same order. The definition also leads to the basic operations in the relational model – namely, selection and projection – since data is always viewed in a row-column tabular format. This view has also driven the models of storage such as row-store and column-store. The key challenges in a relational data model arises when: (1) different occurrences (rows) of an entity require different data layouts (attributes), (2) some occurrences (rows) of an entity have more than one value at the intersection of a row/column while the majority of rows have only one, (3) different occurrences (rows) of an entity are engaged in different kinds of relationships with others entities, (4) there is a need to run queries without meta-data knowledge**.**

The imposition of a homogeneous layout of every row in a relation requires in turn very meticulous, rigid and expensive design efforts. It impedes required changes after the database is in production and makes them expensive. While no data model can guarantee complete data independence, there is a need for a data model that is more flexible than relational.

1.2 Technology and Novelty of Work

Cellular Model: In the cellular data model, the basic logical unit of information is called a cell and is defined as:

"*Elementary Data Cell is a set of four elements comprised of an Occurrence Identifier(O), Entity Identifier(E), Attribute Identifier(A) and Attribute Value(V) i.e. EDC = {O,E,A,V}".*

This basic unit of information is a single cell element. A 'cell-set' can be described as a set of data cells with the same 'E' and 'O'. This representation does not force one to view the data in the form of a "row" or "column", although the cells can be grouped and viewed in a tabular fashion. This model allows multiple values for a given attribute of a cell-set of an entity. Each cell-set of an entity can contain a different set of attributes. Values can be searched without explicit knowledge of the meta-data.

1.3 Applications of Presented System

- *Scenario 1*: Where the corporation's knowledge management application requirements are constantly changing.
- *Scenario 2*: Where the database search requirements cannot be known at design time, or where flexible and powerful ad hoc searches are required. Infobionics does not require extensive and costly up-front schema design; allows both metadata and raw data to be easily searched, and allows both structured data and unstructured data to be managed, navigated and queried.
- *Scenario 3*: Where both technical and end users want a consolidated view of the organization's metadata across the corporation. Infobionics provides a database foundation for a dynamic corporate metadata management system which can be the missing link that transforms corporate data from silos to a truly distributed corporate asset.

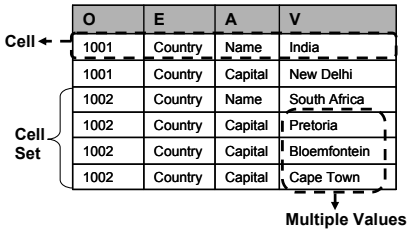

Figure 1. Cellular Model

2. SYSTEM ARCHITECTURE OVERVIEW

Figure 2 illustrates the system architecture. Applications communicate with the server through an API that allows queries to be issued in the Cellular Query Language (CQL). Currently, CQL supports a large portion of SQL syntax. This syntax, while retaining the meaning of the query, operates on cellular datasets as opposed to tables. CQL also provides additional syntax to support queries suited for the cellular model. The CQL statements are parsed to check the syntax and validate the semantics, after which they are processed and allowed to interact with efficient indexed data stores through a set of cellular database primitives. A detailed description of the system and CQL is beyond the scope of this paper.

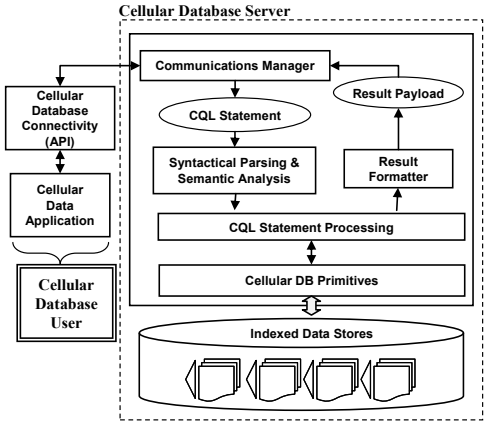

Figure 2. System Architecture of Cellular Database Server

3. DEMONSTRATION SCENARIO

Consider the following piece of text that was written to explain how data can be modeled and queried in a cellular database.

"**Bill Murray** who owns a *Ferrari* and *Porsche* made a cell phone call to a **woman** when lives in New York and Washington D.C. This **woman** had sent an e-mail to **Gary Ford** and also chatted with **Melinda Robert** from Seattle. **Melinda Robert** who owns a *Lincoln* had e-mailed **Gary Ford** asking him to make a phone call to **Bill Murray**. **Robert Hooke** from Lincoln, Nebraska received e-mails from **Bill Murray** and **Gary Ford** to call **Melinda**."

The bold fonts represent 'Person', the italicized represent 'Car' and underlined represent 'Location' for easy identification purposes. The Cellular database does not provide natural language tools to identify such entities, but helps in storage and access of such identified data. Figure 3 illustrates a logical way to store the data in a cellular format. The 'O' values are system generated. It should be noted that all data are represented in a homogenous format of cells. The woman whose name is not given can be represented without an explicit need to change the schema and generate an attribute called 'PersonID'. At the same time, multiple values can be accommodated without having to generate another intermediate table. NULL values are not stored as reflected by a lack of existence. It should be noted that Figure 3 is a logical representation of cells and does not reflect actual physical storage schemes that avoid storing redundant 'O', 'E' or 'A' values. Figure 4 illustrates how the same data would be represented in a relational tabular format. Due to space constraints we represent multiple values as a set. However, in reality that would require the construction of a new intermediate table. Notice the explicit generation of an attribute "PersonID" and storage of NULL values in such a tabular format which is avoided in the cellular format.

O	E	A	V
1	Person	FirstName	Bill
1	Person	LastName	Murray
1	Person	OwnsCar	Ferrari
1	Person	OwnsCar	Porsche
1	Person	CalledPerson	*Person:2*
1	Person	SentEmailTo	*Person:5*

O	E	A	V
2	Person	LivesAt	New York
2	Person	LivesAt	Washington
2	Person	SentEmailTo	*Person:3*
2	Person	SentIMTo	*Person:4*

O	E	A	V
3	Person	FirstName	Gary
3	Person	LastName	Ford
3	Person	CalledPerson	*Person:1*
3	Person	SentEmailTo	*Person:5*

O	E	A	V
4	Person	FirstName	Merida
4	Person	LastName	Robert
4	Person	OwnsCar	Lincoln
4	Person	LivesAt	Seattle
4	Person	SentEmailTo	*Person:3*

O	E	A	V
5	Person	FirstName	Robert
5	Person	LastName	Hooke
5	Person	LivesAt	Lincoln
5	Person	CalledPerson	*Person:4*

Figure 3. Logical representation of data in cellular format

Figure 5 presents the schema which forms the basis of most of the statements and queries in table 1. It should be noted that this schema is different from that shown in Figure 3, which was used only to illustrate differences between relational and cellular models. The schema illustrates the new datatype "LINK" which is used to link any two 'cell sets'. This definition does not force every cell-set to have a link. Due to the capability of the cellular model to have multiple-valued attributes, it provides the ability for a cell-set to link to multiple cell-sets which is declared using the word "MULTI". The "$SetId" is the id of a cell-set that is automatically generated by the system.

The focus of Table 1 is to present a sample of the class of statements and queries that can be handled by the cellular database, while trying to highlight some key capabilities. Space constraints prevent a more detailed explanation of the CQL syntax used. The first noticeable change is the creation of 'Dataset' as opposed to 'Table'. The reason for such nomenclature is that cellular views data as a set of cells not necessarily tied to a logical representation such as a table. The insert statement illustrates the capability to insert multiple values. Other key capabilities include the ability to query metadata, issue global queries without explicit mention of meta-data and other queries to retrieve sets of data using the link structure without the necessity to return a result set containing a homogeneous set of attributes. A result-set can contain other entities as a part of the result. The query in English, the CQL equivalent and the key benefits are presented in Table 1.

4. CONCLUSIONS

This paper describes a next generation database illustrating some key capabilities with a demonstration scenario.

5. REFERENCES

[1] Codd, E.F., "A Relational Model of Data for Large Shared Data Banks", Communications of the ACM 13 (6), 1970.

Acknowledgements: The authors would like to thank and appreciate, inventor of cellular technology, Boris Gelfand for his technical contributions and input on the material. We would also like to thank the whole Infobionics development team for their inputs and feedback.

PersonID	FirstName	LastName	OwnsCar	LivesAt	CalledPerson	SentEmailTo	SentIMTo
1	Bill	Murray	{Ferrari,Porsche}		2	5	
2				{NewYork,Washington}		3	4
3	Gary	Ford			1	5	
4	Melinda	Roberts	Lincoln	Seattle		3	
5	Robert	Hooke		Lincoln	4		

Figure 4. Tabular representation of data[1]

```
Person(
 FirstName STRING,
 LastName STRING,
 OwnsCar SET_LINK MULTI,
 LivesAt SET_LINK MULTI,
 PlacedCallTo SET_LINK MULTI,
 SentEmailTo SET_LINK MULTI,
 Sent_IM_To SET_LINK MULTI,
 SentAnyCommunicationTo SET_LINK MULTI,
);
```

```
Car (
  Make STRING,
  Model STRING,
  VIN STRING UNIQUE
);
```

```
Location(
  City STRING,
  State STRING(2),
  Residents SET_LINK MULTI
);
```

Figure 5. Schema used for queries in table 1

Table 1. Sample of Demonstration Statements and Queries

English	CQL	CQL Benefit
DDL		
Create dataset Person	CREATE DATASET Person(FirstName STRING, LastName STRING, OwnsCar SET_LINK MULTI, LivesAt SET_LINK MULTI, CalledPerson SET_LINK MULTI, SentEmailTo SET_LINK MULTI);	CREATE DATASET is analogous to CREATE TABLE in RDBMS, However, Cellular DBMS supports MULTI declaration and SET_LINK data type.
Insert data into Person	INSERT INTO Person VALUES(NULL, NULL, NULL, {"Washington, DC", "New York, NY"}, "Gary Ford", "Melinda Roberts");	INSERT statement supports insertion of multiple values. Schema of Person dataset is different than that mentioned in Figure 5 for this particular case.
DML		
Create Links between sets	UPDATE Person SET PlacedCallTo = (SELECT $SetId FROM Person P2 WHERE P2.FirstName = "Bill");	Example of UPDATE statement that links two persons together dynamically.
Data Queries Using Links		
Who has placed a phone call?	SELECT FirstName, LastName FROM Person WHERE PlacedCallTo IS NOT NULL;	Use links to easily determine the state of an entity.
What are the total counts for the communication types?	SELECT COUNT(PlacedCallTo) "numCalls", COUNT(SentEmailTo) "numEmails", COUNT(Sent_IM_To) "numIMs", FROM Person;	Simple count of the links tell us what the extent is of the related sets.
Who received a call or message from Washington?	RETRIEVE Receiver.FirstName, Receiver.LastName FROM Location LINK TO Person Initiator LINK TO Person Receiver USING SentAnyCommunicationTo WHERE City = "Washington";	Links provide a more logical and powerful means of querying the database.
What messages were sent from Nebraska to Washington state	RETRIEVE Initiator.FirstName, Initiator.LastName, Receiver.FirstName, Receiver.LastName FROM Location FromLoc LINK TO Person Initiator LINK TO Person Receiver USING SentAnyCommunicationTo LINK TO Location ToLoc	Links provide a more logical and powerful means of querying the database.
Find who has three degrees of separation from Bill Murray.	RETRIEVE Person.FirstName, Person.LastName, FirstDegree.FirstName, FirstDegree.LastName, SecondDegree.FirstName, SecondDegree.LastName, ThirdDegree.FirstName, ThirdDegree.LastName FROM Person LINK TO Person FirstDegree USING SentAnyCommunicationTo LINK TO Person SecondDegree USING SentAnyCommunicationTo LINK TO Person ThirdDegree USING SentAnyCommunicationTo WHERE Person.LastName = "Murray";	Links make it possible to perform queries that are virtual impossible in RDBMs.
Metadata Queries		
Get Attributes that are Multi Value	SELECT $Attributes.Name FROM $Datasets LINK TO $Attributes USING HasAttributes LINK TO $Constraints WHERE $Datasets.Name = "Personr" AND $Constraints.Cardinality <> 1;	Built-in Meta model makes it easy to find dataset attributes that can have multiple values.
Find all places in the database were the word "Lincoln" is used	RETRIEVE $Datasets.Name , $Attributes.Name, COUNT(Value) "How often the value is used" FROM $Datasets LINK TO $Attributes USING HasAttributes LINK TO $Cells WHERE Value = "Lincoln";	Built-in Meta model makes it easy to perform complexity queries involving both data and metadata.
Global Queries		
Give me all the sets where "Lincoln" is used	getSets((SELECT CellLink FROM $Cells WHERE Value = "Lincoln"));	Can search the entire database without explicit knowledge on metadata or data.

[1] For lack of space multiple values are represented as set inside a row-column intersection. Ideally, there would be a need for another 'Car' table that has a primary key –foreign key relationship with 'Persons' table.