Search Efficient Representation of Healthcare Data based on the HL7 RIM

Razan Paul
Department of Computer Science and Engineering,
Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh
Email: razanpaul@yahoo.com

Abu Sayed Md. Latiful Hoque
Department of Computer Science and Engineering,
Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh
Email: asmlatifulhoque@cse.buet.ac.bd

Abstract— **The Reference Information Model (RIM) of Health Level Seven (HL7) standard is a conceptual information model for health care. Data of HL7 RIM Observation class are sparse, high dimensional, and require frequent schema change. Entity Attribute Value (EAV) is the widely used solution to handle these above challenges of medical data, but EAV is not search efficient for knowledge extraction. In this paper, we have proposed a search efficient data model: Optimized Entity Attribute Value (OEAV) for physical representation of medical data as alternative of widely used EAV model. We have implemented EAV or OEAV individually to model RIM Observation class and used relational model for the remaining RIM classes. We have shown that OEAV is dramatically search efficient and occupy less storage space compared to EAV.**

Index Terms— **EAV, OEAV, Open Schema, Sparse Data, Healthcare, HL7, RIM**

I. INTRODUCTION

A number of exciting research challenges posed by health care data are described in [1] [2]. These challenges make health care data different from the data in other industries. Data sparseness arises because doctors perform only a few different clinical lab tests among thousands of test attributes for a patient over his lifetime. It requires frequent schema change because healthcare data need to accommodate new laboratory tests, diseases being invented every day. For the above reasons, relational data representation requires too many columns and hence the data are high dimensional. Summing up continuous valued medical data does not yield any meaning. Many to many relationship between patient and diagnosis requires complex data modeling features. Query incurs a large performance penalty if the data records are very wide but only a few columns are used in the query [3].

HL7 RIM [4] is independent of any implementation technologies, addresses unique challenges of medical data, and broadly covers all aspects of an organization's clinical and administrative information in abstract manner. A RIM-based physical model is suitable as a model of clinical data warehouse. EAV is the widely used solution to handle the challenges of data representation of clinical data warehouse. However, EAV suffers from higher storage requirement and not search efficient. In this paper, we have proposed a search efficient open schema data model: OEAV to convert HL7 RIM abstract information model to physical model. The OEAV is also storage efficient compared to existing EAV model.

Section 2 describes the related work. The overview of EAV and the details organizational structure and analysis of OEAV are given in section 3. A data transformation is required to adopt the existing data suitable for data warehouse representation for knowledge extraction. The transformation is elaborated in section 4. Physical Representation of HL7 RIM is elaborated in section 5. Analytical details of performance of the proposed model is given in section 6. Section 7 offers the result and discussion. Section 8 is the conclusion.

II. RELATED WORK

The RIM's act-centered view [5] of healthcare is based on the assumption that any profession or business including healthcare, consists primarily of a series of intentional actions. The Observation class of RIM captures most of the clinical related act including vital signs, lab test results, allergies, and diagnoses. Data captured by Observation class are sparse, high dimensional and need frequent schema change. For these characteristics of observation data, in [2] authors use EAV [6] to make physical implementation of HL7 RIM Observation class. Moreover, in non-standard medical database designs, EAV is a widely used solution to handle the challenges of observation data, but EAV is not a search efficient data model for knowledge discovery. Moreover, using different data tables for different data types in observation table, instead of using data transformation to make a unified view of data for several data types, they complicates the further knowledge discovery operations. The EAV model for phenotype data management has been given in [7]. Use of EAV for medical observation data is also found in [8] [9] [10]. None of these works is based on HL7 RIM. The suitability of HL7 RIM to represent medical information

doi:10.4304/jcp.5.12.1810-1818

in data warehouse is described in [11]. In [12], authors have built a centralized index of heterogeneous clinical data according to the act-centered view of healthcare from HL7 RIM. The Entity-Relationship Model is proposed in [13]. Agarwal et al. [14] describe a number of different methods for the efficient computation of multidimensional aggregates. In [15] authors propose a relational aggregation operator named data cube generalizing group-by, crosstab, and subtotals.

For storing clinical and genetic data, in [16] authors have proposed bioinformatics system known as the BiolAD-DB system. This Informatics System is publicly available for download and use. To model medical observation data, this system has used EAV to handle challenges posed by health care data. XML based HL7 RIM message is used to communicate with each other by various healthcare systems. In [17], authors have introduced the problem of keyword based information discovery in XML based HL7 RIM message. This paper addresses neither physical representation of observation data nor keyword based information discovery in physical representation of observation data. In [18], authors have proposed a HL7 ontology based framework for medical data transmission in a decentralized environment. This system is a medical multi-agent system. It does not concern how medical observation data are represented physically in secondary storage. In [19], authors have proposed an agent based framework for interoperability among heterogeneous electronic health record systems using Hl7 message. The framework uses HL7 RIM based messages for medical data transmission from one heterogeneous EHR system to another heterogeneous EHR system. This paper does not elaborate either physical or logical representation of heterogeneous electronic health records in secondary storage.

III. Open Schema Data models

We require an open schema data model for HL7 RIM observation class to support dynamic schema change, sparse data, and high dimensionality of observation data. In open schema data models, logical model of data is stored as data rather than as schema, so changes to the logical model can be made without changing the schema.

A. *Entity-Attribute-Value (EAV)*

EAV is an open schema data model, which is suitable for high dimensional and sparse data like medical data. In EAV, every fact is conceptually stored in a table, with three sets of columns: entity, an attribute, and a value for that attribute. In this design, one row actually stores a single fact. It eliminates sparse data to reduce database size and allows changing set of attributes. Moreover, EAV can represent high dimensional data, which cannot be modeled by relational model because existing RDBMS only support a limited number of columns. EAV gives us extreme flexibility but it is not search efficient as it keeps attribute name as data in attribute column and has no tracking of how data are stored.

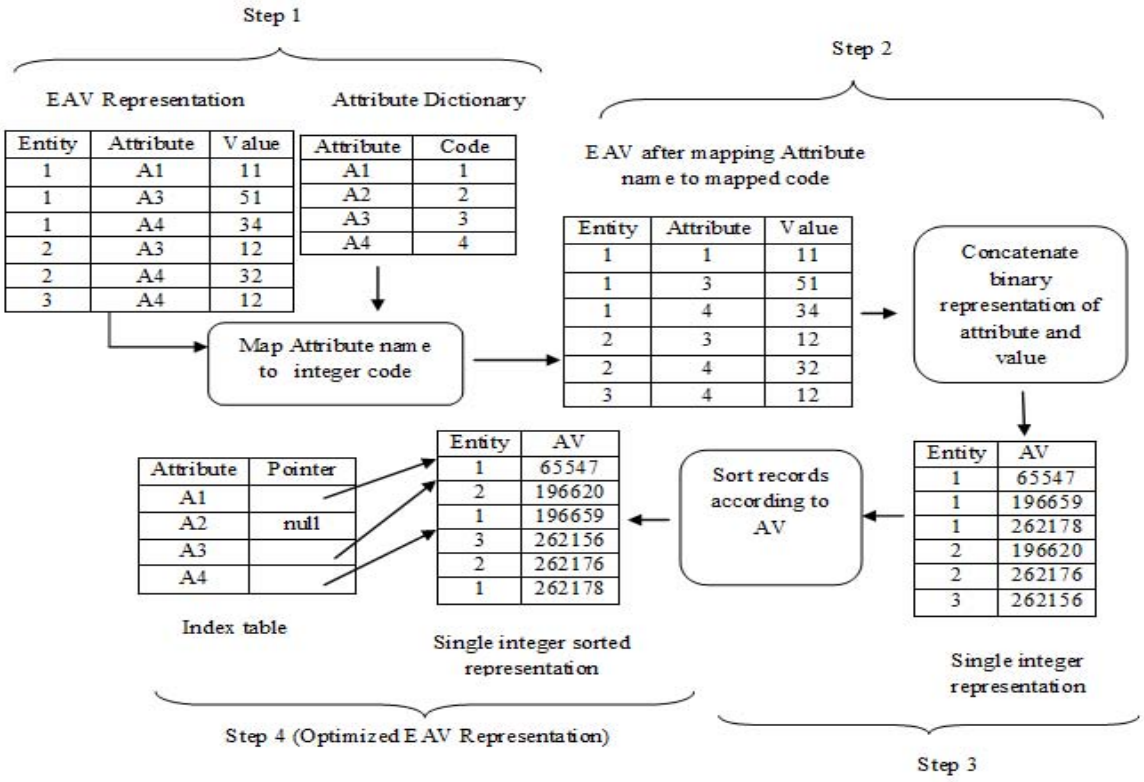

Figure 1. Transformation of EAV model to Optimized EAV (OEAV) model.

B. Optimized Entity Attribute Value (OEAV)

To remove the search inefficiency problem of EAV whilst preserving its efficiency of representing high dimensional and sparse data, we have developed a search efficient open schema data model OEAV. This model keeps data in a search efficient way.

This approach is a read-optimized representation whereas the EAV approach is write-optimized. Most of the data warehouse systems write once and read many times, so the proposed approach can serve the practical requirement of data warehouse. Figure 1 shows the step by step approach of transformation of an EAV data representation to an equivalent OEAV data representation. In step 1, this model constructs an attribute dictionary where there is an integer code for each attribute.

Attribute name of each fact is mapped to an integer code using the attribute dictionary. All types of values are treated as integer using a data transformation as discussed in the following section. In step2, a compact single integer Attribute Value (AV) is created by concatenating binary representation of attribute code and value. In OEAV, every fact is conceptually stored in a table with two columns: the entity and the AV. It maps attribute code and value to p bit and q bit integer and concatenate them to n bit integer AV. For example, an attribute value pair (A3, 51), the code of attribute A3 is 3, will be converted in the following ways: (A3, 51) → (3, 51) → (0000000000000011, 0000000000110011) → 00000000000000110000000000110011 = 196659.

In step 3, the records of optimized EAV are stored as sorted order of AV field. As data are stored in sorted order of AV and the first p bits of AV are for attribute code, the records of an attribute in OEAV table remains consecutively. In step 4, an index structure, which is a part of OEAV representation, is created to contain the starting record number (Pointer) of each attribute in OEAV table. This makes the data partitioned attribute wise, which is expected by most analytical program. In sorted AV field, the values of an attribute also remain in sorted order and binary search can be applied on it. This model constructs a modified B+ tree index on entity field of OEAV to make entity wise search efficient. Here each leaf of the modified B+ tree keeps the block address of attribute values for each entity. These Search efficiencies of OEAV are absent in conventional EAV representation.

IV. Data Transformation Using Domain Dictionary And Rule Base

For knowledge discovery, the medical data have to be transformed into a suitable transaction format to discover knowledge. We have addressed the problem of mapping complex medical data to items using domain dictionary and rule base. The medical data of diagnoses, laboratory results, allergies, vital signs, and etc are types of categorical, continuous numerical data, Boolean, interval, percentage, fraction and ratio. Medical domain expert have the knowledge of how to map ranges of numerical data for each attribute to a series of items. For example, there are certain conventions to consider a person is young, adult, or elder with respect to age. A set of rules is created for each continuous numerical attribute using the knowledge of medical domain experts. A rule engine is used to map continuous numerical data to items using these developed rules.

Data, for which medical domain expert knowledge is not applicable, we have used domain dictionary approach to transform these data to numerical forms. As cardinality of attributes except continuous numeric data are not high in medical domain, these attribute values are mapped integer values using medical domain dictionaries. Here the mapping process as shown in Figure 2 is divided in two phases. Phase 1: a rule base is constructed based on the knowledge of medical domain experts and dictionaries are constructed for attributes where domain expert knowledge is not applicable, Phase 2: attribute values are mapped to integer values using the corresponding rule base and the dictionaries.

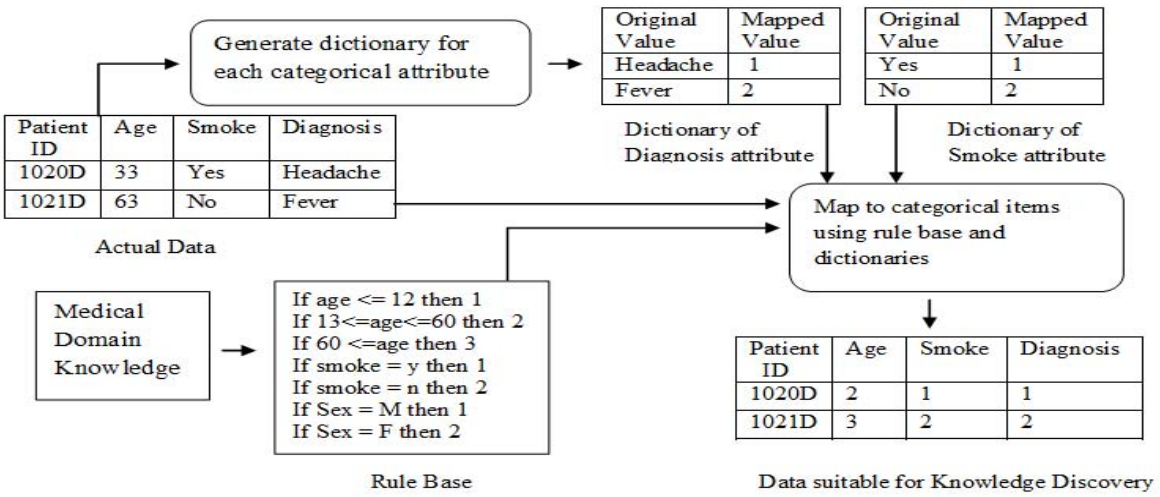

Figure 2. Data Transformation of Medical Data.

V. PHYSICAL REPRESENTATION OF HL7 RIM

All latest HL7 RIM specifications are in a form independent from specific representation and implementation technologies. To make physical representation of HL7 RIM from its abstract model, we need to make the physical implementation of following things: data types, classes, aggregations, inheritance structures, and associations.

A. Implementing Data Type

HL7 has abstract semantic data type specification for its openness towards representation and implementation technologies [20]. HL7 RIM has a number of data types as shown in table I, which need to be implemented in physical database context. We have mapped primitive data types to standard database types. To implement coded data types [21], we have made a separate table for each coded type and have kept a foreign key named codeID in class table, which refer to the primary key of coded type table. For example, to implement CD data type, a separate table is created with the following column: (codeID BIGINT, code ST, codeSystem UUID, displayName ST). Code system table holds all the information regarding coding system. To implement each compound data type, we have kept multiple columns in the class table, where each column is for each part of compound data type. These columns are used as part of the class table. For example, Encapsulated Data (ED) is implemented with two columns: Media Type CS, DATA CLOB.

Each complex type has been implemented according to each one's structure. For example, entity name type specializes LIST<ENXP>. Entity Name is implemented as separate table with following columns: (ID, Valid-Time, and Formatted). For entity name parts, LIST<ENXP> is implemented as a separate table where each row contains name part info and a foreign key refers to the row, for which this name part info belongs to, of Entity Name table. Name Part info includes the following columns: ReferenceID INT, Namepart ST, PartType CS , Qualifier Comma separated string.

To implement collection data type, we have created a separate table per RIM class for each collection data type where each row contains a value from multi values and a foreign key to the row, for which this value belongs to, of parent table. For example, IVL<int> in class ACT for attribute Repeat Number is implemented in the following way: we have created a new table named ACT-IVL-Repeat-Number for class ACT where each row contains an integer value and a foreign key, which refers to the corresponding row of table ACT.

B. Modeling RIM Classes

As data captured by Observation class is sparse, high dimensional and need schema change, we have modeled Observation class using EAV and OEAV data models individually to see what performs better. All the remaining RIM classes have been implemented using relational model. We have made physical representation of each RIM base class using a relation that includes all the attributes of the base class.

TABLE I.
HL7 Data Types

Category of HL7 Data type	HL7 data types of this category	Description
Primitive	BL, BN, ST, II, TN, INT, REAL, and TS	HL7 specification throws away all of the standard variable types and invents its own (Gives new name)
Coded	CS, CV, CE, CO and CD	Coded types are the HL7 Vocabulary Domains types, which exposed several enums in a number of domains by a variety of codes.
Compound	ED, SC, TEL, RTO, PQ, MO, UVP, and PPD	Compound data types allow a collection of data of different types to be grouped into a single object.
Complex	EN, AD, PN, ON, and NPPD	A composite and derivative of other existing data types
Collection	BAG, LIST, SET, IVL, and HIST	Collection Data Type is a multi values data type. It allows a collection of data of the same types to be grouped into a single object.
Special	PIVL, EIVL, and GTS	These types required special processing logic.

C. Implementing HL7 RIM Inheritance

The physical implementation of RIM inheritance can be made in one of the following ways: a single table per entire class hierarchy, map each concrete class to its own table that contains inherited attributes, and map each class to its own table that does not contain inherited attributes. With the exception of the Observation class, which we discuss later, we have used the third strategy to model RIM inheritance structure. In this strategy, one table is created per class with one column per class attribute and necessary identification information.

D. Implementing Associations and Aggregation

We have mapped RIM associations using foreign keys. To implement a one-to-many association, we have implemented a foreign key from the "one table" to the "many table". One-to-one relationship is implemented in the same way. To make physical representation of each association of type many to many, a table have been

created, which keeps the combination of the primary keys of the two tables, which are in many-to-many relationship with each other. In RIM, only aggregation is Act Relationship. It has been implemented as a separate table that includes primary key consists of two fields: ID of inbound act and ID of outbound act with the other Act Relationship object attributes in the table.

E. Physical Representation of Observation Class

Observation statements describe what was done, what was actually observed, and assertive statements. The data type of value attribute of Observation class is ANY, which means it can be any valid HL7 data type. To make observation data suitable for knowledge discovery, we have transformed value attribute using domain dictionaries and rule engine to make a unified view of data, integer representation, for several data types.

Observation class using EAV approach: We have implemented observation class using "map each concrete class to a separate table that does not contain inherited attributes" approach with one exception that is Code attributes of Act class is implemented in observation table. To implement in this way, we have kept the following attributes: Value, Code, Interpretation Code, method code, TargetSiteCode and a key to keep the reference with the act entity in the observation table using EAV data model. Here, every type of code in HL7 RIM is varchar (64). Here ID, Value is the entity, value of EAV model and Code represents attribute of EAV. ID refers to ID of ACT class of HL7 RIM.

Obsevation class using EAV
• ID BIGINT • Code Varchar(64) • Value Varchar(64) • InterpretationCode Varchar(64) • MethodCode VarChar(64) • TargetSiteCode Varchar(64)

Obsevation class using OEAV
• ID BIGINT • AV 32bit INT • InterpretationCode 32bit INT • MethodCode 32bit INT • TargetSiteCode 32bit INT

Figure 3. Physical representation of Observation Class using different data models.

Observation class using OEAV approach: Here observation class has been implemented in the same way as EAV approach. The difference is that Value and Code attributes have been implemented as AV. Therefore, we have kept the attributes: AV, InterpretationCode, MethodCode, TargetSiteCode, and a key to keep the reference with the act entity in the observation table using OEAV data model. Here AV field is the combination of Observation Code and the Observation Value. A modified B+ tree has been created based on patient ID where leaf node of the tree contains the block address of each attribute value of the particular patient. In OEAV, InterpretationCode, MethodCode, and TargetSiteCode are converted into 16-bit integer using domain dictionaries. Each Code represent an attribute and is transformed into 16-bit integer using attribute dictionary. Value is transformed into 16-bit integer using domain dictionaries and Rule base. The compact single integer Attribute Value (AV) is created by concatenating binary representation of 16-bit attribute code and 16-bit value. Here AV is 32-bit integer and ID refers to ID of ACT class of HL7 RIM.

VI. ANALYSIS OF EAV & OEAV

Let b be the total number of blocks of observation table and k is the total number of attributes of observation table.

A. Analysis of Storage Capacity of EAV

Let n = total number of facts, $q =$ average length of attribute names, $g =$ average length of values. In EAV, 32 bits (4 bytes) is required to represent entity. Size of each fact in EAV in bytes is

$$(4+q+g)\ldots\ldots\ldots(1)$$

Hence, the total size to hold all facts in bytes is

$$n\times(4+q+g)\ldots\ldots\ldots(2)$$

B. Space Complexity of Medical domain dictionaries and Rule Base

Let $C_i =$ cardinality of ith attribute where domain expert knowledge is not applicable, $L_i =$ average length of ith attribute name, P = number of categorical attributes. Codes of attributes are not stored explicitly and the index of attribute is the code. Domain dictionary storage of ith attribute is $C_i \times L_i$ bytes. Total domain dictionaries storage (SD) in bytes is

$$\sum_{i=1}^{p}(C_i\times L_i)\ldots\ldots\ldots(3)$$

If the size of rule base storage is R, the dictionary and rule base storage (SDR) in bytes is

$$\sum_{i=1}^{p}(C_i\times L_i)+R\ldots\ldots\ldots(4)$$

C. Analysis of Storage Capacity of OEAV

Let $p =$ number of attributes, $q =$ average length of attribute names. Total storage of attribute dictionary is $p \times q$ bytes. Let $S =$ size of each block address in byte. Total storage of index table is $p \times q + p \times S$ bytes. In OEAV, 32 bits are required to represent entity and 16 bits are required for attribute and value individually. 64 bits = 8 bytes = size of each fact in OEAV. Let n = total number of facts, m = total number of facts in a block, w = word size (bytes). Total number of blocks is

$$\lceil n/m \rceil\ldots\ldots\ldots(5)$$

The number of words per fact is

$$\lceil 64/w \rceil \ldots\ldots\ldots (6)$$

For block i where $1 \leq i \leq \lceil n/m \rceil$, the number of words per block is

$$\lceil (m \times \lceil 64/w \rceil) \rceil \ldots\ldots\ldots (7)$$

The size of a block in bytes is

$$w(\lceil (m \times \lceil 64/w \rceil) \rceil) \ldots\ldots\ldots (8)$$

Hence the size to hold all facts in bytes is

$$S = \lceil n/m \rceil \times w \times (\lceil (m \times \lceil 64/w \rceil) \rceil) \ldots\ldots\ldots (9)$$

In OEAV, total size to hold all facts in bytes = storage for facts + storage for domain dictionaries and rule base + storage for attribute dictionary + storage for index table + storage for modified B+ tree

$$= \lceil n/m \rceil \times w \times (\lceil (m \times \lceil 64/w \rceil) \rceil) + \sum_{i=1}^{p} (C_i \times L_i)$$

$$+ R + (p \times q) + (p \times q + p \times S) + B \ldots\ldots\ldots (10)$$

D. CUBE Operation

Group by is basis of all OLAP operation. Multiple Group-by operations can form a CUBE operation and each group-by is called a cuboid. Each cube operation requires aggregation of measures at some level. For aggregate operation, we consider only max, min, average, and count. In EAV, for each aggregate operation, it has to scan all the blocks because it keeps attribute name as data in attribute column. In OEAV, a count operation can be computed on an attribute from index table without any block access. For max and min operation on a attribute it has to scan only 1 block because it keeps each attribute data separate and in sorted order. It has to scan b/k blocks to compute average.

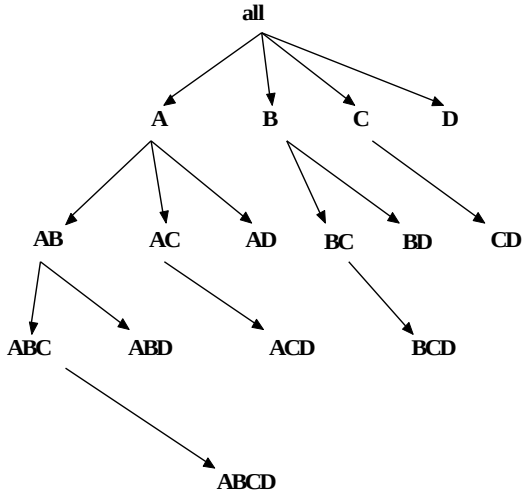

Figure 4. Bottom-Up approach of CUBE computation.

As medical data is sparse, widely used top down approach of CUBE computation [15], is not a good choice. Because top down approach does not consider null data in cube computation, so computing each cuboid from a number of other parent cuboids does not yield the correct result all the time. Considering null data in cube computation can solve this problem but it will add huge complexity in cube computation with open schema data models. It is because that finding out for which entities a particular attribute value is null is time consuming and hard in these open schema data models.

Here we build the CUBE using bottom-up algorithm as in figure 4. It starts building CUBE by computing cuboids on a single attribute, then cuboids on a pair of attributes, then cuboids on three attributes, and so on. Candidate of cuboids are generated from child cuboids. This is the reverse of widely used top down approach of CUBE computation. Here pruning is bases on data representation of open schema data models as open schema data models store sparse data, so the algorithm does not need to perform cube computation on sparse data. The algorithm reduces disk read costs by computing all same-length cuboids in one scan of DB. The algorithm uses hashing technique [22] to compute a cuboid. In EAV, facts are not clustered attribute wise and have no entity index, so every search becomes full scan of all blocks. In OEAV, facts are clustered attribute wise and have entity index, it requires scanning only necessary attribute partitions and values.

VII. RESULTS AND DISCUSSION

The experiments were done using PC with core 2 duo processor with a clock rate of 1.8 GHz and 3GB of main memory. The operating system was Microsoft Vista and implementation language was c#. We have designed a data generator that generates all categories of random data: ratio, interval, decimal, integer, percentage etc. This data set is generated with 5000 attributes and 5-10 attributes per transaction on average. We have used highly skewed attributes in all performance evaluations to measure the performance improvement of our proposed open schema data model in worst case. For all performance measurement except storage performance, we have used 1 million transactions.

A. Storage Performance

Figure 5 shows the storage space required by EAV and OEAV. The EAV occupies significantly higher amount of storage than OEAV. This is due to the data redundancy of EAV model.

B. Time Comparison of Projection Operations

Figure 6 shows the performance of projection operations on various combinations of attributes. Almost same time is needed with different number of attributes in EAV, as it has to scan all the blocks whatever the number of attributes. In OEAV, it can be observed that the time requirement is proportional to the number of attributes projected. This is because that the query needs to scan more number of blocks as the number of attributes increases.

C. Time Comparison of Select Queries

Figure 7 shows the performance of multiple predicates select queries on various combinations of attributes.

Figure 7 shows almost same time is taken with different number of attributes in EAV as it has to scans all the blocks twice whatever the number of attributes in predicate. The graph shows how time is varied in OEAV with different number of attributes as it scans number of attribute partitions proportional to number of attributes in select quires. This experiment shows EAV has taken much higher time compared to OEAV. It is because it has no tracking of how data are stored, so it has to scans all the blocks once to select entities and has to scan all the blocks one more time to retrieve the attribute values for the selected entities. OEAV has taken the lower time as it does not need to read unused attributes to select entities and can retrieve attribute values of these entity without reading any unused attribute value using entity indexing .

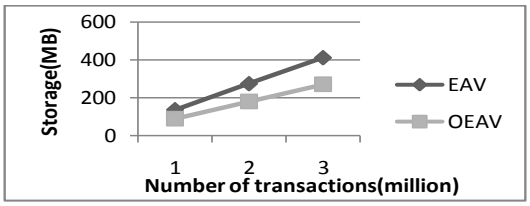

Figure 5. Storage performance.

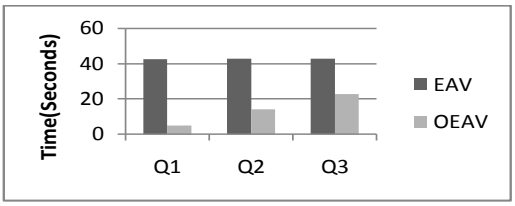

Figure 6. Time comparison of projection operations.

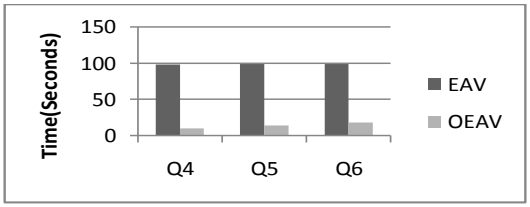

Figure 7. Time comparison of select queries.

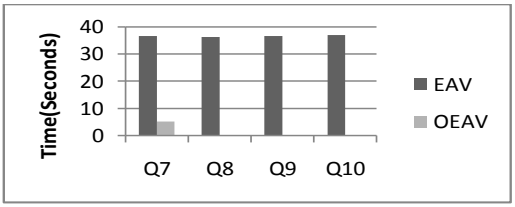

Figure 8. Time comparison of aggregate operations.

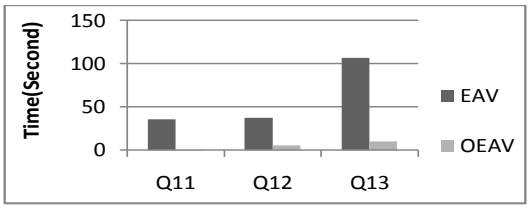

Figure 9. Time comparison of statistical operations.

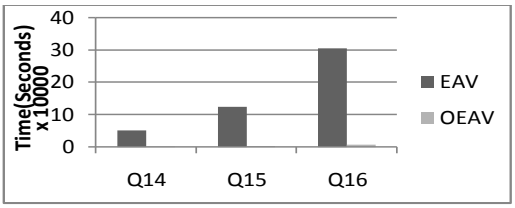

Figure 10. Time comparison of CUBE operations.

Q1: Select A_i from observation;
Q2: Select A_i , A_j ,A_k from observation.
Q3: Select A_i, A_j, A_k, A_l, A_m from observation.
Q4: Select * from observation where A_i='XXX'.
Q5: Select * from observation where A_i='XXX' AND A_j='YYY'.
Q6: Select * from observation where A_i='XXX' AND A_j='YYY' AND A_k='ZZZ'.
Q7: Select AVG (A_i) from observation.
Q8: Select Max (A_i) from observation.
Q9: Select Min (A_i) from observation.
Q10: Select Count (A_i) from observation.
Q11: Select Median (A_i) from observation.
Q12: Select Mode (A_i) from observation.
Q13: Select Standard Deviation (A_i) from observation.
Q14: Select A_i, A_j, Max (Am) from observation CUBE-BY (A_i, A_j)
Q15: Select A_i, A_j, A_K, Max (A_m) from observation CUBE-BY (A_i, A_j, A_k)
Q16: Select A_i, A_j, A_K, A_m, Max (A_n) from observation CUBE-BY (Ai, A_j, A_k, A_m)

D. Time Comparison of Aggregate Operations

Aggregate operations compute a single value by taking a collection of values as input. Figure 6 shows the performance of various aggregate operations on a single attribute. Time is not varied significantly from one aggregate operation to another as different aggregate operations need same number of data block access for most of the cases. Figure 8 shows EAV has taken much higher time than OEAV as it has to scan all the blocks to compute each operation. OEAV has taken negligible time for max, min, count operations on a single attribute as to find max and min it has to scan only 1 block and count result is computed from its index table. For average operation on an attribute, it has taken considerable time, as it has to scan all the blocks of that attribute.

E. Time Comparison of Statistical Operations

Figure 9 shows the performance of various statistical operations on a single attribute. Time is varied

significantly from one statistical operation to another as different statistical operations need different sorts of processing. This experiment shows EAV has taken much higher time compared to OEAV. It is because it has no tracking of how data are stored, so it has to scan all the blocks to compute each operation. We can see from this figure OEAV has taken negligible time for median operation as it has to scan 1 or 2 blocks for this operation. For mode and standard deviation, it has to scan all data blocks of the attribute for which particular operation is executing once, twice respectively.

F. Time Comparison of CUBE Operations

The CUBE operation is the n-dimensional generalization of group-by operator. The cube operator unifies several common and popular concepts: aggregates, group by, roll-ups and drill-downs and, cross tabs. Here no pre-computation is done for aggregates at various levels and on various combinations of attributes. Figure 10 shows the performance of CUBE operations on various combinations of attributes. It can be observed that the number of attributes in cube operations leads to the time taken as CUBE operation computes group-bys corresponding to all possible combinations of CUBE attributes. The experiment results show that EAV has taken much higher time compared to OEAV as it does not partition data attribute wise and it has no entity index.

The next data set of interest is the Zoo Data Set [23] from UCI Machine Learning Repository, which has the similar characteristics like medical data. It contains 101 instances each described by 18 attributes (included 16 discrete and 2 numerical attributes).We have repeated this dataset 100 times to get 10100 instances for performance evaluation. We have taken an average value from 10 trials for each of the test result. We have achieved a query performance faster in the range of 15 to 70 compared to existing EAV model for this dataset. We have achieved a storage performance of 1.25 times better compared to existing EAV. The results for Zoo data set are almost identical to the result for the synthetic dataset.

TABLE II.
HOW MANY TIMES OEAV IS CHEAPER AND FASTER COMPARED TO EAV

Operation	OEAV
Storage	1.26
Projection	15.15
Selection	56.3
Aggregate	69.54
Statistical	50.45
CUBE	45.95

VIII. CONCLUSION

Search efficient representation of Healthcare data is important for knowledge discovery. Once this knowledge is discovered, the results of such analysis can be used to define new guidelines for improving medical care and treatment. The Observation class of HL7 RIM represents most of the clinical related act like diagnoses, laboratory results, allergies, vital signs etc. Therefore, most of the knowledge discovery operations will be performed on observation data. Observation class is sparse, high dimensional and need frequent schema change and EAV is a widely used solution to handle these challenges. However, EAV is not a search efficient data model for knowledge discovery. We have proposed a search efficient open schema data model OEAV to handle these challenges as alternative of EAV. Table 2 summarizes how many times OEAV is cheaper and faster compared to EAV. The experiment results show our proposed open schema data model is dramatically efficient in knowledge discovery operation and occupy less storage compared to EAV.

We have proposed the conversion of HL7 RIM abstract information model to physical model, which includes HL7 RIM abstract data type conversion, modeling RIM classes, inheritance, association and aggregation. The solution to address the problem of mapping complex medical data to items has also been proposed here. As our solution is based on industry standard, this approach is widely applicable.

REFERENCES

[1] P. B. Torben and J. S. Christian, "Research Issues in Clinical Data Warehousing," in Proceedings of the 10th International Conference on Scientific and Statistical Database Management , Capri, 1998, p. 43–52.

[2] E. J. Thomas, T. W. Jeffrey, and C. D. Joel, "A health-care data model based on the HL7 reference information model," IBM Systems Journal, vol. 46, no. 1, pp. 5 - 18, 2007.

[3] A. Rakesh, S. Amit, and X. Yirong, "Storage and Querying of E-Commerce Data," in Proceedings of the 27th International Conference on Very Large Data Bases, 2001, pp. 149 - 158.

[4] HL7 Reference Information Model. (n.d.). Retrieved 02 12, 2009, from Health Level Seven International: http://www.hl7.org/Library/data-model/RIM/modelpage_mem.htm

[5] V. Lowell, S. Barry , and C. Werner, (2007). Foundation for the electronic health record: an ontological analysis. Retrieved 02 10, 2009, from Buffalo Ontology: http://ontology.buffalo.edu/medo/HL7_2004.pdf

[6] W. W. Stead, E. W. Hammond, and J. M. Straube, "A chartless record—Is it adequate?," Journal of Medical Systems, vol. 7, no. 2, pp. 103-109, 1983.

[7] J. Li, M. Li, H. Deng, P. Duffy, and H. Deng, "PhD: a web database application for phenotype data management," Oxford Bioinformatics, vol. 21, no. 16, pp. 3443-3444, 2005.

[8] J. Anhøj, "Generic design of Web-based clinical databases," Journal of Medical Internet Research, vol. 5, no. 4, p. 27, 2003.

[9] C. Brandt, A. Deshpande, and C. Lu, "TrialDB: A Web-based Clinical Study Data Management System AMIA 2003 Open Source Expo," in Proceedings of the American Medical Informatics Association Annual Symposium, Washington, 2003, p. 794.

[10] P. M. Nadkarni et al., "Managing Attribute—Value Clinical Trials Data Using the ACT/DB Client—Server

Database System," The Journal of the American Medical Informatics Association, vol. 5, no. 2, p. 139–151, 1998.

[11] L. A. Jason et al., "Mapping From a Clinical Data Warehouse to the HL7 Reference Information Model," in AMIA Annu Symp Proc, 2003, p. 920.

[12] A. Jiye, L. Xudong, D. Huilong, L. Haomin, and J. Peipei, "An Act Indexing Information Model for Clinical Data Integration," in Bioinformatics and Biomedical Engineering, 2007. ICBBE 2007. The 1st International Conference on, 2007, pp. 1099 - 1102.

[13] C. P. Peter, "The entity-relationship model—toward a unified view of data," ACM Transactions on Database Systems, vol. 1, no. 1, pp. 9 - 36, 1976.

[14] S. Agarwal et al., "On the Computation of Multidimensional Aggregates," in Proceedings of the 22th International Conference on Very Large Data Bases, 1996, pp. 506-521.

[15] G. Jim, C. Surajit, and B. Adam, "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals," Data Mining and Knowledge Discovery, vol. 1, no. 1, p. 29–53, 1997.

[16] D. A. Nielsen, M. Leidner, C. Haynes, M. Krauthammer, and M.J. Kreek, "The BiolAD-DB System - An Informatics System for Clinical and Genetic Data," Molecular Diagnosis & Therapy: 11, 15-19 (2007).

[17] V. Hristidis, F. Farfán, R. Burke, A. Rossi and J. White, "Challenges for Information Discovery on Electronic Medical Records," Florida International University Technical Report. Feb 2007.

[18] B. Orguna and J.Vub, "HL7 ontology and mobile agents for interoperability in heterogeneous medical information systems," Computers in Biology and Medicine, Special Issue on Medical Ontologies, Volume 36, Issues 7-8, July-August 2006, Pages 817-836.

[19] F. Cao, N. Archer and S. Poehlman, "An Agent-based Knowledge Management Framework for Electronic Health Record Interoperability," Journal Of Emerging Technologies In Web Intelligence, vol. 1, no. 2, pp. 119-128, November 2009.

[20] Data Types - Abstract Specification. (2004, 11 29). Retrieved 02 05, 2009, from Health Level Seven International: http://www.hl7.org/v3ballot/html/infrastructure/datatypes/datatypes.htm

[21] HL7 Vocabulary Domains. (n.d.). Retrieved 02 05, 2009, from Health Level Seven International: https://www.hl7.org/library/data-model/RIM/C30202/vocabulary.htm

[22] G. Graefe, "Query Evaluation Techniques for Large Databases," ACM Comp. Surveys, vol. 25, no. 2, pp. 73-170, June 1993.

[23] Zoo Data Set. (n.d.). Retrieved 03 01, 2010, from Machine Learning Repository: http://archive.ics.uci.edu/ml/support/Zoo