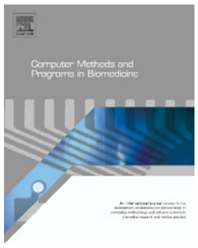
COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE 110 (2013) 99–109

journal homepage: www.intl.elsevierhealth.com/journals/cmpb



Alternatives to relational database: Comparison of NoSQL and XML approaches for clinical data storage

Ken Ka-Yin Lee[a,*], *Wai-Choi Tang*[b,1], *Kup-Sze Choi*[a,2]

[a] *Centre for Integrative Digital Health, School of Nursing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*
[b] *Information Technology Committee, Hong Kong Doctors Union, Hong Kong*

ARTICLE INFO

Article history:
Received 13 April 2012
Received in revised form
24 October 2012
Accepted 31 October 2012

Keywords:
Clinical data
Relational database
NoSQL
XML
Information storage and retrieval

ABSTRACT

Clinical data are dynamic in nature, often arranged hierarchically and stored as free text and numbers. Effective management of clinical data and the transformation of the data into structured format for data analysis are therefore challenging issues in electronic health records development. Despite the popularity of relational databases, the scalability of the NoSQL database model and the document-centric data structure of XML databases appear to be promising features for effective clinical data management. In this paper, three database approaches – NoSQL, XML-enabled and native XML – are investigated to evaluate their suitability for structured clinical data. The database query performance is reported, together with our experience in the databases development. The results show that NoSQL database is the best choice for query speed, whereas XML databases are advantageous in terms of scalability, flexibility and extensibility, which are essential to cope with the characteristics of clinical data. While NoSQL and XML technologies are relatively new compared to the conventional relational database, both of them demonstrate potential to become a key database technology for clinical data management as the technology further advances.

1. Introduction

Clinical data is dynamic, sporadic, and heterogeneous in nature [1]. While they share some of the characteristics of the data managed by conventional data warehouse, special attention is required in the design of database schema because of the unique features they possess. Currently, storage of clinical data largely relies on relational database management systems. The relational database model is the most common and a proven approach to store and query data in various forms [2]. However, the major drawback is the need to pre-design the exact field structures of the data, which is required in the process of database normalization to ensure data consistency [3]. In addition, the relational database model is not practical for certain forms of data that require a lot of fields to handle different types of data involved, where most of the data fields are indeed left unused due to the nature of the data. A relational database storing these kinds of data will contain many empty fields, resulting in inefficient storage and poor performance. Medical data, especially clinical notes, are such an example. To deal with these issues, we attempt to use a class of database known as NoSQL and extensible markup language (XML) to develop databases that can cope with the special features of clinical data more effectively.

* *Corresponding author at*: Centre for Integrative Digital Health, School of Nursing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. Tel.: +852 2766 6399.
E-mail addresses: kenleeky@gmail.com (K.K.-Y. Lee), tang.eric233@gmail.com (W.-C. Tang), kschoi@ieee.org (K.-S. Choi).
[1] Tel.: +852 2423 8293.
[2] Tel.: +852 3400 3214.

0169-2607/$ – see front matter © 2012 Elsevier Ireland Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cmpb.2012.10.018

To determine the suitable database approaches for the storage of structured clinical data, the performance of databases developed using NoSQL and XML are compared in this paper. The database approaches concerned are NoSQL database, XML-enable database and native XML database. The comparisons are made from the following aspects: query performance, scalability, flexibility and extensibility. The databases are populated with de-identified real data obtained from the clinic of a private general practitioner (GP) in Hong Kong. The rest of the paper is organized as follows. Section 2 gives an overview on the application of relational databases, the NoSQL database movement and XML databases for storing medical data. Section 3 describes the data structure used for clinical data management and the types of database approaches to be investigated in the paper. Section 4 discusses the three database approaches concerned based on the experimental results and our experience in the development. Finally, a conclusion is given in Section 5.

2. Backgrounds

It is generally agreed by many researchers that the storage and management of medical data is a difficult task. As pointed out by Cios and Moore [4], medical data are voluminous and complex in nature. They are the "most rewarding and difficult" data to analyze. The wide variety of medical data is indeed one of the major obstacles to the implementation of electronic health records (EHR) in hospitals [5]. Medical data are generated from multiple sources, e.g. laboratory or pharmacy, which resides in different databases of different data structures. They also exist in different forms, both semi-structured and structured formats. The heterogeneity makes integrated access of medical data a challenging issue [6]. Therefore, an effective approach for storing medical data and facilitating data analysis is needed.

A number of models have been proposed for medical data storage. Los [7] modified the conventional row-modelling database to accommodate the heterogeneous nature of medical data. Prather et al. [8] proposed a new knowledge discovery method for medical data in order to find the relationships between medical concepts and their related properties in a large clinical database. Rector et al. [9] emphasized the importance of the context of medical data and the relationships between different data sets. Nevertheless, most medical data storage systems are built based on relational database approach, which is not efficient and flexible enough to handle the data.

There is a constant demand for alternatives to the relational database. Recently, the movement towards the "NoSQL" is gaining attention. It refers to a class of data storage systems that is significantly different from the conventional relational database. For example, NoSQL does not require pre-defined schema, relationships and keys. Database table join queries are also not supported [10]. The NoSQL technology is inspired by the Web 2.0 developers and communities who realize that relational database is not suitable for the management of real-time social networking websites where the data are voluminous and heterogeneous. NoSQL databases are low-cost, schema-free, and horizontally scalable to accompany new computing resources when the need arises. In general, NoSQL uses key-value stores, BigTable implementation, document store, and graph databases which are uncommon in traditional relational database design [11]. As the nature of the data in Web 2.0 applications shares similarities with that of clinical data, a few attempts to apply the NoSQL approaches, e.g. using the key-value stores, have been made for clinical data storage [12,13], although the research effort is at an early stage. Despite a relatively new technology, NoSQL is significant in that it is a common database approach for cloud computing and thus a promising solution for cloud-based clinical systems.

On the other hand, XML is a well-known and robust markup language designed to encode documents electronically and represent the data in a structured way. It has been demonstrated widely that XML can be used for storing structured data with a high degree of flexibility. For example, a database architecture is proposed for medical data by exploiting XML's strength in data interoperability and application integration [14]. An XML-based record system is also developed to provide customizable storage of medical data based on the needs of patients [15]. Medical markup language (MML), a variation of XML, is used to design hospital information system for medical data exchange [16]. In these applications, medical data are represented using XML and stored as XML files. It is worth noting that popular standards for clinical records (e.g. EN-ISO13606, OpenEHR, HL7 RIM-CDA) also use XML to codify information.

Furthermore, XML databases are proposed and developed for efficient processing of structured XML data. Jagadish et al. proposed an XML database to optimize XML-based queries and processing [17]. Meier also proposed an XML database to store, index and query a large collection of XML data [18]. Both of these XML databases are considered "native" which refers to the characteristic that the internal model of the databases depends on XML and uses XML documents as the fundamental unit of storage. Alternatively, traditional relational databases can also be exploited to take advantage of the benefits of XML, where XML data is mapped to a conventional relational database [19]. Databases of this kind are known as XML-enabled databases. Few XML databases have been developed for medical data storage [20–22], and they model the data at the document level rather than the content level inside an XML document. Note that XML databases are often also classified as a type of NoSQL database [10].

Standardized and normalized data reference models are needed to represent clinical information. One of the benefits of clinical data models is that they support the level of complexity required to correctly interpret the information with context. The data models also facilitate the processing of semantic meaning within the data, which is of growing interest in EHR development. To achieve normalized clinical data models, a strategy known as the two-level methodology is proposed to decouple knowledge models from system design [23]. The methodology provides flexibility for the integration of different knowledge models with the clinical systems while the integration can be independent from the system design. This strategy is being adopted by an increasing number of systems. Examples of normalized reference models to represent the clinical information includes EN-ISO13606, OpenEHR, and HL7 RIM-CDA [24].

In this paper, we investigate the performance of database approaches in storing clinical data using standard clinical models. The three database approaches – namely, NoSQL database, XML-enabled database and native XML database – are compared using real clinical data to identify their strengths and weaknesses. The results will provide an indication on the database approaches that are appropriate for clinical data and, in a broader sense, for the design and implementation of EHR systems.

3. Method

3.1. *Data structure for clinical data using HL7-CDA*

Use of normalized reference model to represent clinical information is important to enhance correct interpretation of clinical findings, interoperability with other systems as well as compatibility with research data. For this study, HL7-CDA is adopted as a reference model as it is one of the most common models for clinical systems [25]. As depicted in Fig. 1, sample clinical data is shown inside the HL7-CDA header and body. The personal and demographic information are placed in the header section of the HL7-CDA while the detailed clinical notes such as symptoms and problems are coded in the HL7-CDA body section.

The standard HL7-CDA specification provides a concise template to store pre-defined data such as name, date, time, and codified information. However, storing the actual clinical notes turn out to be more problematic, as they are usually stored as free-text without pre-defined fields. Usually the data is a mixture of both descriptive items and the associated parametric values [26]. To properly process the data and to make them searchable, it is necessary to format the information with a computer-processable structure and convert them into a structured format [27]. Raw clinical data can be structured with tree data structure [28]. Fig. 2 is a generalized representation of the tree-structured clinical data.

The figure depicts the basic organization of the clinical data. The main purpose of this arrangement is to hierarchically enclose a list of concepts at the top level, allowing sub-concepts to attach to the main concept in a flexible way. The free-text clinical notes field is at the top level of each entry. A clinical notes entry contains medical concepts, with numeric or textual properties. This arrangement offers maximum flexibility to include as many medical concepts as required for a record. Each concept can have any number of properties, in textual or numerical form. It is important to allow the use of numeric values which are particularly useful for statistical analysis and facilitate data processing by other software. Other specialized fields of different data types, e.g. date, time, lists and binary objects, can be incorporated conveniently with this arrangement.

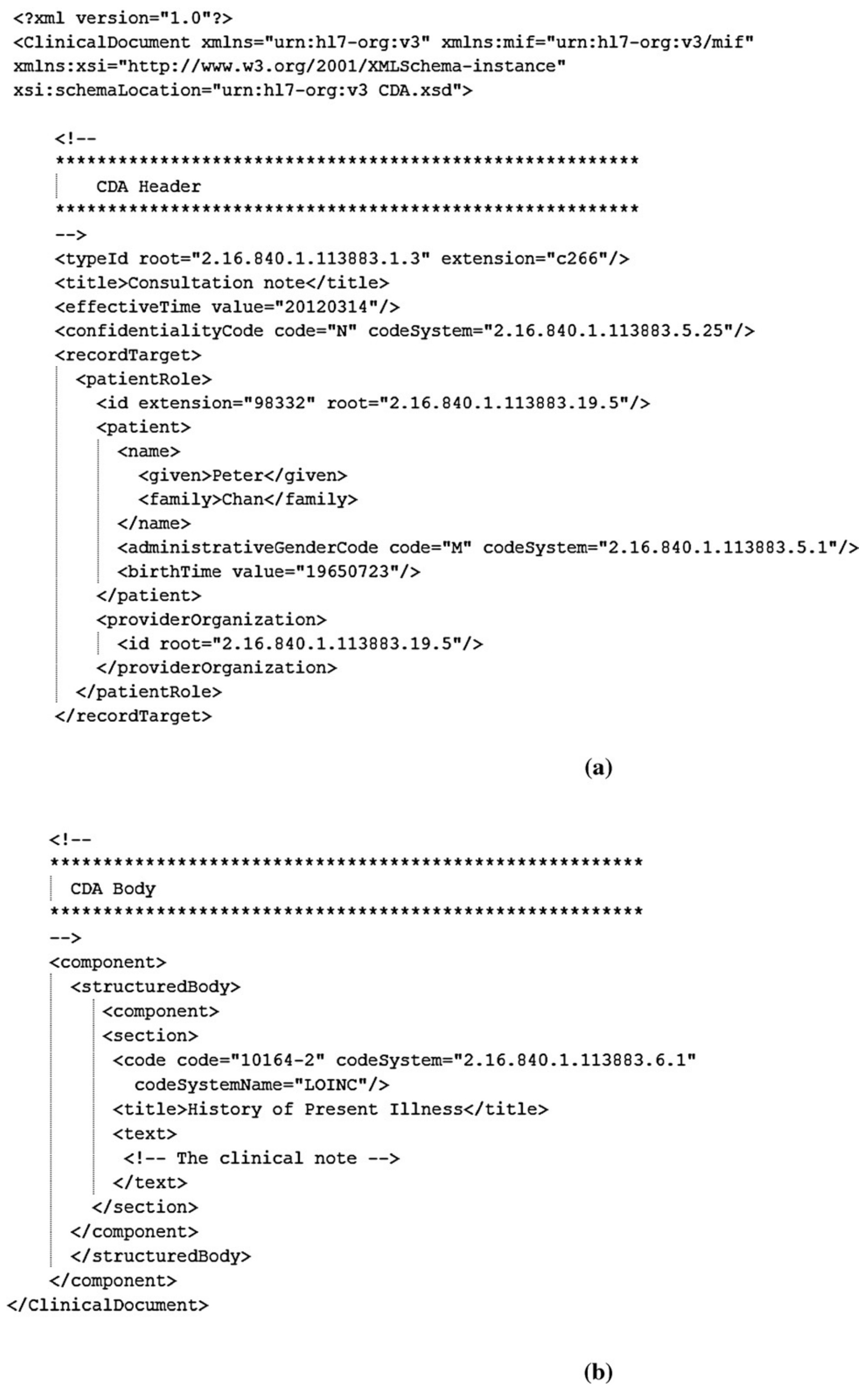
To adhere to the two-level methodology of separating knowledge models from system designs, a knowledge-level structured data definition (also known as archetypes [23]) is implemented to model the detailed clinical notes at the free-text level. The clinical information is arranged and structured according to the data structure shown in Fig. 2. The model is also compatible with HL7-CDA structure. The semantic description of the data definition is defined in Fig. 3. This definition is utilized for the three database approaches concerned for the purpose of this study.

3.2. *NoSQL schema-less database modelling*

Although NoSQL is an attractive alternative to relational databases, the NoSQL databases (e.g. the Apache Cassandra) are still new, not as mature and production-tested as relational databases. As a result, many database developers of mission-critical systems would rather adopt new designs with existing relational database software, and there are various examples illustrating the use of schema-less data on a relational database [29]. In this study, the NoSQL schema-less approach is implemented by using the Microsoft SQL Server 2008 (MS-SQL). Despite a relational database that uses the SQL language to query data, MS-SQL can implement schema-less database concepts and designs. It is also an industry standard which is well-known and readily available for the research. In order to use relational database with these database models, it is still necessary to create a database schema by breaking down the raw data into individual tables/columns and inserted them into one or multiple database tables. Different schema-less database design approaches can be used to implement the structured clinical data defined in section 3.1. They include, but not limited to, one concept per table (column-based, fixed multiple columns), fixed column name (column-based, non-fixed-rows key-value pair), fixed key-value pair and generalized key-value pair [30]. The implementation of these four approaches is illustrated in Fig. 4.

The one-concept-per-table approach is a simple schema-less data model. As shown in Fig. 4(a), all concepts are broken down into individual tables and in their normal forms. This approach is obviously impractical due to the inherent nature of medical record which could involve a huge amount of common medical concepts. A large number of tables are thus required to develop a database, which is unmanageable even for a small clinic setting. The fixed-column-name approach illustrated in Fig. 4(b) is limited to a very specific scope of a medical record. All the medical concepts are pre-defined in a database table as columns, and the capacity is limited by the maximum number of columns allowed by the database system. The scalability of this method is therefore problematic as the number of columns required for representing medical concepts can easily increase beyond the limit of any database systems. Also, it is hard to pre-determine the number of columns due to the variety of data in medical records.

Key-value stores are a common technique in NoSQL schema-less database design. Databases implemented with the fixed-key-value-pair approach are able to provide more flexibility. Refer to Fig. 4(c), the values are stored as key-value pairs, making it flexible to include as many concepts as necessary. However, scalability remains an issue as the number of columns available from a database can be readily exhausted by the amount of medical concepts required for creating clinical notes. Besides, it is computationally intensive to process a query which retrieves records with a particular concept since individual medical concepts can be located at any of the key fields (e.g. Key1 and Key2 in the figure). The database engine



```
<?xml version="1.0"?>
<ClinicalDocument xmlns="urn:hl7-org:v3" xmlns:mif="urn:hl7-org:v3/mif"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hl7-org:v3 CDA.xsd">

    <!--
    ********************************************************
        CDA Header
    ********************************************************
    -->
    <typeId root="2.16.840.1.113883.1.3" extension="c266"/>
    <title>Consultation note</title>
    <effectiveTime value="20120314"/>
    <confidentialityCode code="N" codeSystem="2.16.840.1.113883.5.25"/>
    <recordTarget>
      <patientRole>
        <id extension="98332" root="2.16.840.1.113883.19.5"/>
        <patient>
          <name>
            <given>Peter</given>
            <family>Chan</family>
          </name>
          <administrativeGenderCode code="M" codeSystem="2.16.840.1.113883.5.1"/>
          <birthTime value="19650723"/>
        </patient>
        <providerOrganization>
          <id root="2.16.840.1.113883.19.5"/>
        </providerOrganization>
      </patientRole>
    </recordTarget>
```

(a)

```
    <!--
    ********************************************************
      CDA Body
    ********************************************************
    -->
    <component>
      <structuredBody>
         <component>
         <section>
          <code code="10164-2" codeSystem="2.16.840.1.113883.6.1"
            codeSystemName="LOINC"/>
          <title>History of Present Illness</title>
          <text>
           <!-- The clinical note -->
          </text>
         </section>
      </component>
      </structuredBody>
    </component>
</ClinicalDocument>
```

(b)

Fig. 1 – A sample HL7-CDA document with (a) header and (b) body.

is required to perform an exhaustive search for the concepts concerned by visiting all the key fields.

The flexibility of databases developed with the generalized-key-value-pair approach is better than those built using the last two approaches, with virtually unlimited rows to work with. In this approach, each individual concept is represented as a single row in the database table, as shown in Fig. 4(d). The major drawback, however, is that the number of rows can grow considerably even for a modest number of records. For example, for a medical database containing 50,000 records where one record is created per clinic visit, if the average number of concepts per record is ten, 500,000 rows

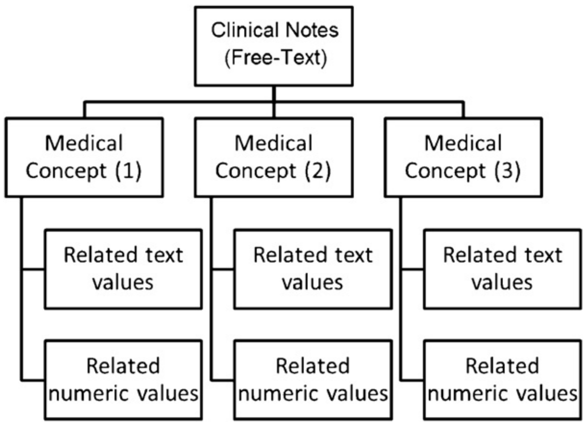


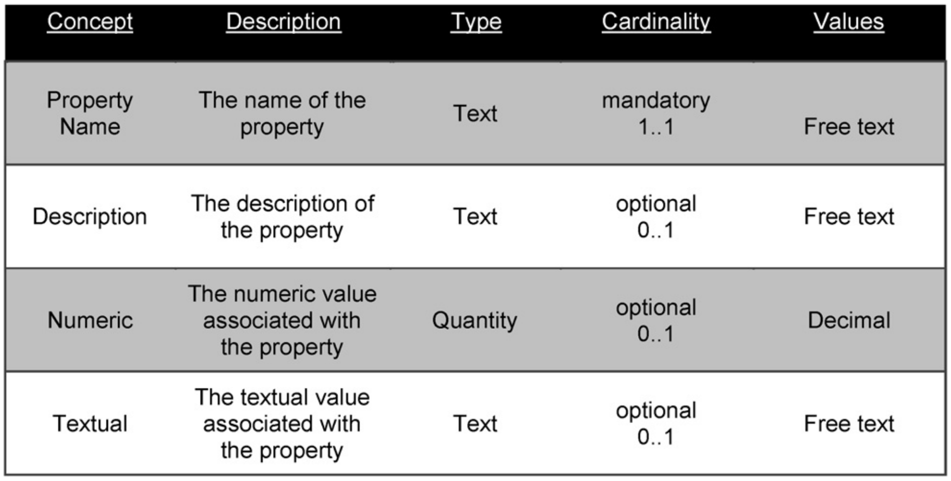
Fig. 2 – A general data structure for clinical notes.

will be required to record all the visits. Further, performing queries on such databases is very complicated, making data extraction a challenging task. Among these four schema-less approaches, the generalized-key-value-pair approach is the best one that is most suitable for the development of schema-less database for structured clinical data, concerning the level of scalability and usability it is able to offer. In the paper, the generalized-key-value-pair approach is therefore compared to the XML-enabled database and the native XML database.

3.3. *XML database modelling*

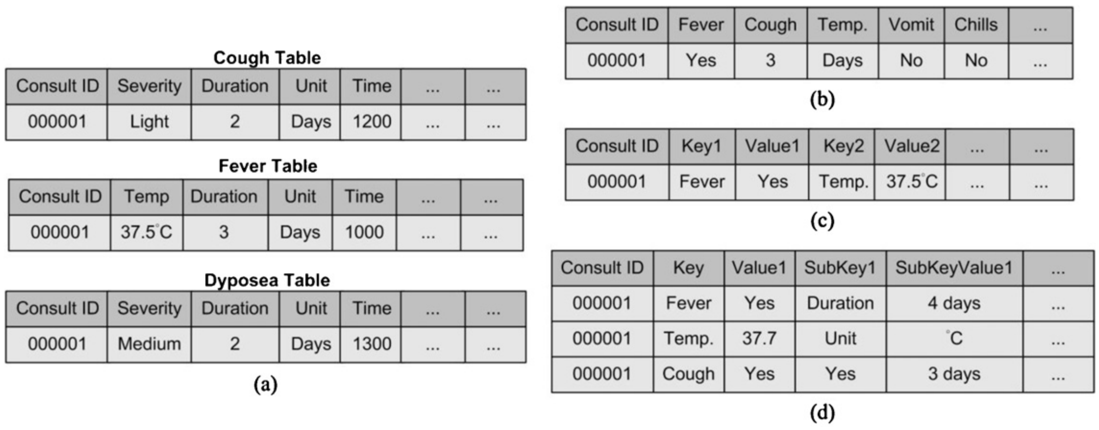
XML natively supports data storage in a hierarchical manner. It is relatively easier to implement the structured clinical data using XML. The basic XML data structure used to handle the structured clinical data discussed previously in section 3.1 is shown in Fig. 5. At the highest level is the element "RecordList" that, as it is named, contains a list of medical records. At the next level is the standardized HL7-CDA "Record" which contains the individual items of the records. Each record may contain patient demographic information such as name and gender, stored using HL7-CDA data formats. This is optional and can be added or removed as needed. At the level further below is the content of the clinical notes, i.e. a list of medical concepts.

Each record may contain zero or more medical concepts. The name of each concept is represented in the element "Name". Each medical concept may associate with zero or more related properties. For each property, there may be zero or more textual attributes which are denoted as "Text";



Concept	Description	Type	Cardinality	Values
Property Name	The name of the property	Text	mandatory 1..1	Free text
Description	The description of the property	Text	optional 0..1	Free text
Numeric	The numeric value associated with the property	Quantity	optional 0..1	Decimal
Textual	The textual value associated with the property	Text	optional 0..1	Free text

Fig. 3 – Semantic description of knowledge-level data definition.



Cough Table

Consult ID	Severity	Duration	Unit	Time	...	...
000001	Light	2	Days	1200	...	...

Fever Table

Consult ID	Temp	Duration	Unit	Time	...	...
000001	37.5˚C	3	Days	1000	...	...

Dyposea Table

Consult ID	Severity	Duration	Unit	Time	...	...
000001	Medium	2	Days	1300	...	...

(a)

Consult ID	Fever	Cough	Temp.	Vomit	Chills	...
000001	Yes	3	Days	No	No	...

(b)

Consult ID	Key1	Value1	Key2	Value2	...	...
000001	Fever	Yes	Temp.	37.5˚C	...	...

(c)

Consult ID	Key	Value1	SubKey1	SubKeyValue1	...
000001	Fever	Yes	Duration	4 days	...
000001	Temp.	37.7	Unit	˚C	...
000001	Cough	Yes	Yes	3 days	...

(d)

Fig. 4 – NoSQL database modelling: (a) one concept per table (three concepts shown), (b) fixed column name, (c) fixed key-value pair, and (d) generalized key-value pair.

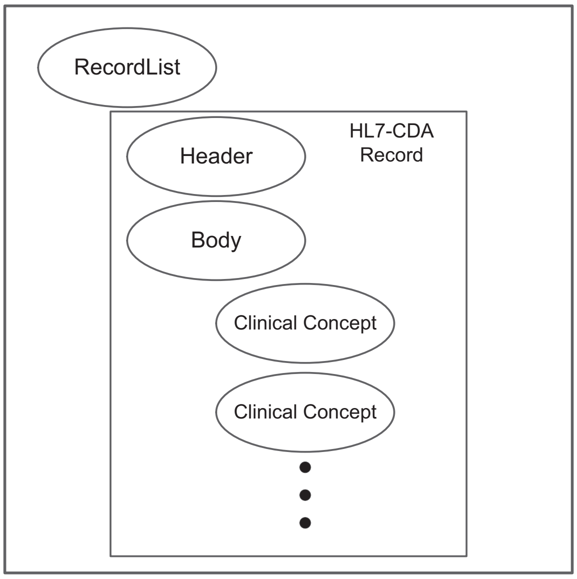


Fig. 5 – The XML data structure.

and zero or more numeric attributes which are denoted as "Numeric". These two attributes are optional, and either one or both of them can be used to represent the concept's properties as necessary.

By utilizing XML, the individual clinical record can be stored inside a single XML field of the database. The resulting database can be readily scaled up to cater for a large number of concepts and their associated properties, which provides a flexible and consistent way for the storage of clinical records.

There are two different XML database implementations for the XML schema described above, namely, the XML-enabled database and the native XML database. In this study, the MS-SQL is used to develop an XML-enabled database for the schema. The XML data type in MS-SQL is used to handle XML data. It can be used with other conventional database fields in the same data row (see Fig. 6(a)). Additionally, XML indexes are also added to enhance query performance. A total of 4 indexes are added, including XML Primary Index, XML Secondary Index – PATH, XML Secondary Index – PROPERTY and XML Secondary Index – VALUE. On the other hand, the open source database management system eXist (version 1.4.1) is used to develop the native XML database in this study due its performance in handling medical records as reported in previous work [22]. The native XML database

Consult ID	XML field	...
000001	<Data>	...
000002	<Data>	...
000003	<Data>	...

(a)

XML file list
record000001.xml
record000002.xml
record000003.xml

(b)

Fig. 6 – (a) XML data are stored with an individual field in XML-enabled database, (b) XML files containing clinical data are stored as a list of files in native XML database.

[Start]Symptom name [Separator] Attribute [Separator] Attribute [Separator] … [End]

(a)

{Fever| Temperature |37.7 Celsius} {Cough| Duration | 4 Days} {Disease (Endocrine) | Glucose | Diabetes Mellitus (Asymptomatic)|}

(b)

Fig. 7 – Electronic clinical note: (a) format, (b) a sample.

uses the same XML schema to represent the clinical data. Standard indexes enabled by the default database installation are utilized, including Structural, Range, N-Gram, and Full Text. While the data are stored using the XML field in the XML-enabled database, they are stored as an actual XML file in the native XML database, as shown in Fig. 6.

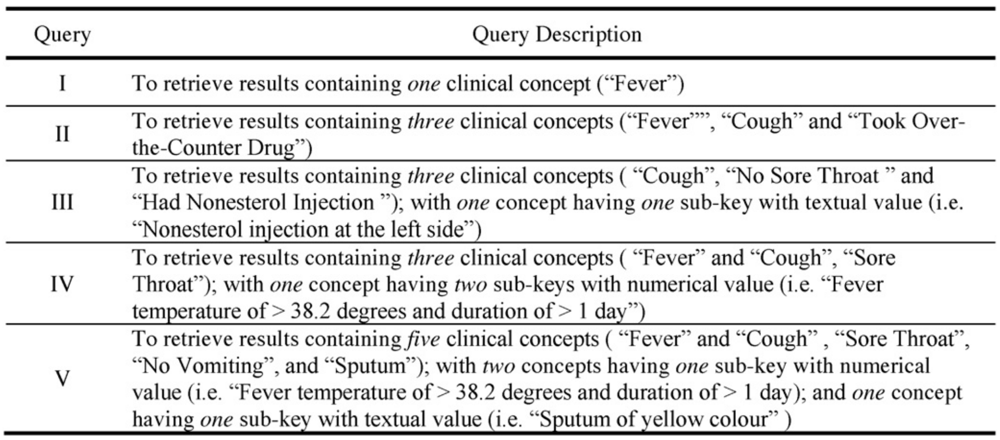
4. Results and discussions

In this section, the performance of the generalized-key-value-pair NoSQL database, XML-enabled database and the native XML database, denoted as approach I, approach II and approach III respectively, are discussed based on our experience in the database development and their applications for the storage of structured clinical data, from the perspectives of query time, data preparation efficiency, scalability, extensibility and flexibility.

4.1. *Test data*

The performance of the three database approaches is evaluated by using real clinical notes obtained from a GP's clinic. The data contain 50,000 consultation records of general family medicine nature (about 120 MB of text). The types of diseases include, but not limited to common cold and influenza. Additionally, associated with these records, data about drug intake and vaccination (when applicable) are also included. The data are de-identified to remove protected health information. The clinical notes are stored electronically using a clinical database with the format shown in Fig. 7(a). A clinical note begins with the marker [Start], followed by the symptom name (concept), and the attributes associated with the symptom. The symptom name and the attributes that follow are separated by the [Separator] markers. The clinical note ends with the [End] marker. A sample of the clinical notes represented in this format is shown in Fig. 7(b).

The data is transformed into the HL7-CDA structure (as described in Section 3.1) through a series of mapping and text manipulation processes. Firstly, the standard values (e.g. patient name, patient ID, birth date, consultation date and time, health provider information, etc.) are mapped to the corresponding field in the CDA document. Next, the clinical note details (as shown in Fig. 7) is taken apart, analyzed and arranged in a tree structure, and subsequently re-organized according to the knowledge-level data definition as described in Section 3.1. The transformation is achieved with dedicated



Query	Query Description
I	To retrieve results containing *one* clinical concept (“Fever”)
II	To retrieve results containing *three* clinical concepts (“Fever””, “Cough” and “Took Over-the-Counter Drug”)
III	To retrieve results containing *three* clinical concepts (“Cough”, “No Sore Throat ” and “Had Nonesterol Injection ”); with *one* concept having *one* sub-key with textual value (i.e. “Nonesterol injection at the left side”)
IV	To retrieve results containing *three* clinical concepts (“Fever” and “Cough”, “Sore Throat”); with *one* concept having *two* sub-keys with numerical value (i.e. “Fever temperature of > 38.2 degrees and duration of > 1 day”)
V	To retrieve results containing *five* clinical concepts (“Fever” and “Cough” , “Sore Throat”, “No Vomiting”, and “Sputum”); with *two* concepts having *one* sub-key with numerical value (i.e. “Fever temperature of > 38.2 degrees and duration of > 1 day); and *one* concept having *one* sub-key with textual value (i.e. “Sputum of yellow colour”)

Fig. 8 – Queries with different complexity.

parser scripts developed using C#. A script is created to identify the standard fields in the original dataset, which are then stored in an XML file to become the header section of the CDA document. Furthermore, for the section of clinical note details, the text is processed by another script that reads through the text from the beginning to the end of the section. When the script encounters a clinical concept, it will mark the starting location and then continue to read through until reaching the end of the concept. The parameters in between the concept are noted in the process. The whole concept (and its parameters) is subsequently re-written in an XML file with a structure based on the knowledge-level data definition. Finally, the standard values and clinical note details are combined as a CDA document and written to the database.

4.2. *Query time*

Three database systems are developed respectively to implement the database approaches concerned in the study. The systems are built using a personal computer equipped with a 2.53 GHz Intel(R) Core(TM) i5 M540 CPU, 6.00 GB RAM and a 128 GB solid-state drive. The operating system is Microsoft Windows 7. The database implementing the generalized-key-value-pair approach (approach I) and the XML-enabled database (approach II) are both developed using MS-SQL. The native XML database (approach III) is developed with eXist, where X-Path, a query language for XML documents, is also adopted to perform queries against the imported dataset. The query time of the three databases is evaluated by making five different queries with varied complexity as shown Fig. 8. The query time is obtained using the SQL Server Profiler for approaches I and II, and the eXist Admin Client for approach III.

The variation in query time with the size of the database is also studied. For each of the three database approaches, the time taken to make the queries with varying complexity specified above is measured with databases containing 1000, 5000, 10,000 and 50,000 records respectively. At each setting, the query is made for 10 times to calculate the average query time and the standard deviation (SD). The timing performances of the queries are given in Tables 1–5 respectively. The query time for the two XML databases is quite disappointing. As shown in Fig. 9, for the simple queries, the performances of all the three approaches are similar when the size of database is small but diverge significantly as the number of records is increased to 10,000 records and beyond. The performance of approach I (i.e. NoSQL database approach) is significantly faster than that of the two XML approaches for large databases (i.e. 10,000 and 50,000 records). The result is similar for the more complex queries; the query speed of approach I for large database is on average 2–8 times better than the other two XML approaches, even though the NoSQL approach has significantly more data rows to process. This is probably due to the fact that the conventional relational database used to implement the NoSQL approach has been fully optimized by the industry throughout the years of development, whereas XML engines are still quite new and yet to reach the same level of optimization, especially for native XML databases. The XML indexes are helpful to speed up queries but not enough to reach the level that the NoSQL database approach has attained.

4.3. *Data preparation*

From the perspective of data preparation efficiency, database schemas design in the three approaches does not involve the preparatory steps required in conventional relational databases which include the definition of database table headers, data normalization, definition of primary keys and foreign keys and the creation of relationships between tables. Hence, clinical data can be imported directly into the databases “as-is”, without the need to transform the data in order to fit them into relational schemas. Furthermore, approach III is more advantageous than approach I & II since native XML databases provide further convenience by enabling XML files to be processed in its native form without the need to import them into the database tables. This can potentially save a significant amount of time to import large datasets. Hence, the entire relational schema design process in traditional relational databases is not required for NoSQL and XML databases, considerably reducing the effort of database development.

Table 1 – Query performance of the three approaches on database with different size: query I.

Approach	Database implementation	Mean ± SD (ms) – query I			
		1000 records	5000 records	10,000 records	50,000 records
I	Generalized key-value pair	29.30 ± 4.79	61.60 ± 5.76	103.70 ± 11.22	816.30 ± 11.62
II	XML-enabled	174.60 ± 9.58	862.00 ± 17.40	1507.40 ± 19.37	7318.40 ± 40.20
III	Native XML	52.10 ± 7.89	229.70 ± 70.64	497.70 ± 137.77	2795.40 ± 236.91

Table 2 – Query performance of the three approaches on database with different size: query II.

Approach	Database implementation	Mean ± SD (ms) – query II			
		1000 records	5000 records	10,000 records	50,000 records
I	Generalized key-value pair	40.70 ± 4.92	68.70 ± 7.90	104.90 ± 8.16	1192.30 ± 19.57
II	XML-enabled	113.80 ± 3.43	340.20 ± 19.41	614.60 ± 13.48	4350.10 ± 41.60
III	Native XML	76.20 ± 5.41	417.90 ± 15.61	945.90 ± 160.42	5009.70 ± 168.24

Table 3 – Query performance of the three approaches on database with different size: query III.

Approach	Database implementation	Mean ± SD (ms) – query III			
		1000 records	5000 records	10,000 records	50,000 records
I	Generalized key-value pair	36.20 ± 2.78	65.10 ± 6.85	101.30 ± 7.17	1425.50 ± 17.69
II	XML-enabled	137.90 ± 11.28	504.40 ± 14.73	888.60 ± 21.24	4093.80 ± 113.79
III	Native XML	101.10 ± 5.00	608.50 ± 114.87	1296.30 ± 223.86	7505.60 ± 592.60

Table 4 – Query performance of the three approaches on database with different size: query IV.

Approach	Database implementation	Mean ± SD (ms) – query IV			
		1000 records	5000 records	10,000 records	50,000 records
I	Generalized key-value pair	175.10 ± 6.92	283.60 ± 11.35	339.50 ± 16.88	1294.90 ± 8.45
II	XML-enabled	111.30 ± 19.67	380.80 ± 16.65	642.90 ± 23.42	3925 ± 44.25
III	Native XML	149.70 ± 5.60	910.80 ± 39.78	1956.00 ± 273.20	10,384 ± 1220.27

4.4. *Scalability*

Scalability of a database can be interpreted from the perspective of database architecture and deployment, and also from the perceptive of the data type (i.e. clinical notes in this study). Here, we discuss the scalability of the three databases approaches from the latter perspective. Approach I is outstanding among the NoSQL approaches concerned in the paper. Scalability is enabled by allowing additional clinical concepts to be included by adding new rows to the database table. However, as each concept occupies a new row in the table and a general clinical note usually contains a number of concepts, the size of database will grow exponentially and the storage space will be consumed significantly, potentially leading to database administration and maintenance problems, especially for busy clinics or hospitals.

The XML approaches handle clinic notes in a more structured manner. They are able to handle a large amount of text data in a single XML field or file while maintaining the structure of the clinical note. Databases developed with these approaches can be easily scaled up as there is no limitation on data granularity or the level of the hierarchy tree required to represent the clinical notes. Hence, the integrity of data models can be maintained, where one consultation record always occupies one row of database entry, regardless of the amount of clinical notes. The number of rows is thus kept to a minimum. Modern database supporting XML data type/file can accept more than 1 gigabyte of data, which is far beyond the size that a typical textual clinical note would require. Data storage capacity is therefore not be a limiting factor. This is a significant advantage of using XML database for clinical data storage when compared to NoSQL approaches.

Table 5 – Query performance of the three approaches on database with different size: query V.

Approach	Database implementation	Mean ± SD (ms) – query V			
		1000 records	5000 records	10,000 records	50,000 records
I	Generalized key-value pair	293.10 ± 5.76	454.00 ± 16.91	673.50 ± 23.58	2341.70 ± 39.02
II	XML-enabled	105.50 ± 20.03	381.80 ± 18.43	664.30 ± 27.79	3922.80 ± 32.06
III	Native XML	241.30 ± 16.99	1351.80 ± 36.40	2913.60 ± 130.72	16,338 ± 615.95

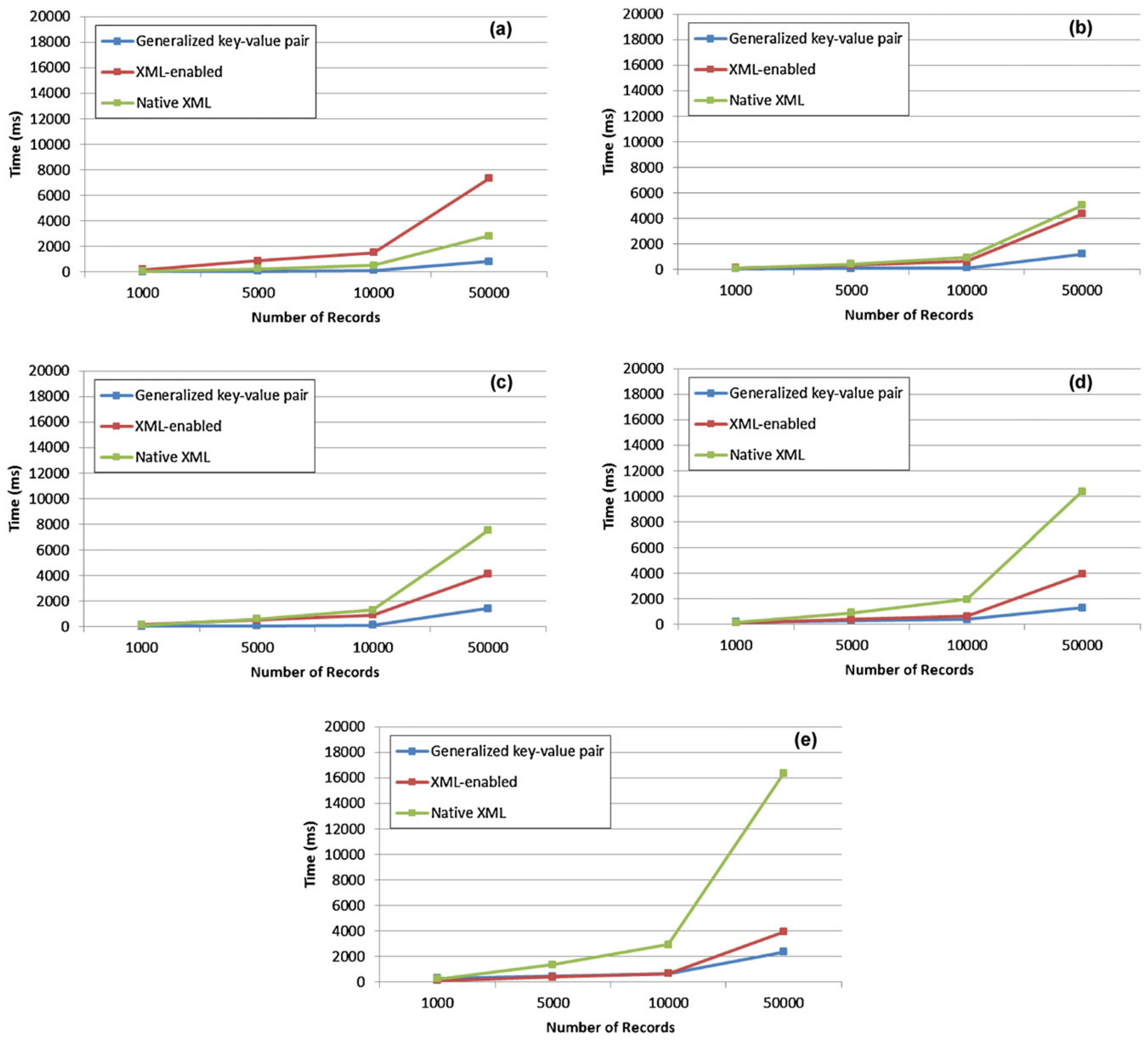


Fig. 9 – Variation in query time with the size of database: (a) query I, (b) query II, (c) query III, (d) query IV, (e) query V.

4.5. *Flexibility*

In conventional relational databases, one must first model the data in the initial phase of the development. Data modelling is restricted here by the permissible number of columns of the database management system. In contrast, both the NoSQL database design and the XML structure used in the XML databases are more flexible in that there is no need to pre-define the required number of columns. Data with complex structures can always be added subsequently. Further, re-design of schema is not required when the content is changed since the schema is generalized for any clinical concepts. In particular, for XML databases containing clinical data represented with XML data types, queries can be performed conveniently with the X-Query language which is sufficient for performing most queries encountered in conventional relational database tables [31]. Between the two XML approaches, it is necessary to write additional SQL codes in approach II in order to import and fit the data into the XML data types supported by MS-SQL. This procedure is not required for native XML database developed with eXist, where raw data are simply imported as files.

4.6. *Extensibility*

The XML file format is portable and platform independent. It is both human readable and machine processable. The format also facilitates logical data management. The original content and integrity of the data in clinical notes can be more closely preserved and represented, without the need to break down the data into different tables/columns which is often required in relational and NoSQL database schemas. Another advantage of the XML approach is the potential interoperability with other systems. For example, the international Health Level Seven messaging standard (including the HL7-CDA specification used in this study) commonly used in health information systems is specified in XML format, and the proposed XML data structure is expected to integrate well with systems developed with this standard with minimal development effort. This is a clear advantage of the XML approaches when compared to the NoSQL approaches.

5. Conclusion

A review of the current approaches of clinical data storage indicates that the NoSQL approach is a viable alternative to relational database design as it provides better query performance while still retaining certain degree of scalability and flexibility. Meanwhile, the use of XML for clinical database development is a promising solution but it is not yet ready for prime-time usage, especially in a large production or time-sensitive environment. The performances of NoSQL database, XML-enabled database and native XML database are compared in this papers by using real data provided by a GP. It is concluded that the implementation of NoSQL approach on a relational database provides software developers with the opportunity to take advantages of schema-less and non-relational design for handling complex clinical data while staying with the familiar and well-established relational database systems. The NoSQL is found to be a flexible approach for handling clinical data but it falls short in terms of scalability and extensibility when compared to the XML approaches.

The XML approaches reduce the preparatory time for populating the database, especially when native XML database is used. It is readily scalable to cope with the complexity of clinical data. XML databases are therefore a potential approach to tackles the issues arising from the use of relational databases for EHR development. The NoSQL approach shared some of these characteristics but not as intuitive as the XML database in handling clinical data.

However, making queries with the XML approaches is found to be less effective compared to the NoSQL approach implemented on a conventional relational database. This is attributed to the fact the conventional relational database is a mature and well-established technology that has undergone thorough optimization by the industry throughout the years. As XML database technology is relatively new, the performance of the XML design could be hindered by the know-hows in implementing the design for large datasets. It also presents a steep learning curve for database developers who have been accustomed to conventional relational database. Further optimization is required to fully exploit the potential of XML database and maximize the performance of the data search engine.

This study attempts to explore the vast opportunities of NoSQL and XML technologies in clinical data management. The prototype system developed is initially tested with a maximum of 50,000 records only. Further evaluation using larger datasets, or even multiple databases and data warehouse, will give more comprehensive results on the performance of the XML and NoSQL databases. Besides, the feasibility of using the XML/NoSQL design in other specialty medicine practices is also worth studying as differences in workflows and medical concepts, usage and management, could affect the performance of the database. Finally, the XML database engine will be further optimized through index optimization, for example, and re-evaluated when new versions of the database software are available. For the NoSQL approaches, future work includes the use of NoSQL databases (instead of MS-SQL in this study), such as key-value storage or document storage database, to develop an NoSQL clinical database and evaluate the performance.

Acknowledgements

This work was supported in part by the Direct Allocation Grant of the Hong Kong Research Grants Council (Project No. A-PJ38) and the Hong Kong Polytechnic University (Project Nos. 1-ZV5E, 1–ZV6C, and 1-87RF).

REFERENCES

[1] S. Wasan, V. Bhatnagar, The impact of data mining techniques on medical diagnostics, Data Science Journal 5 (2006) 119–126.
[2] P. Atzeni, V.D. Antonellis, Relational Database Theory, Benjamin-Cummings Publishing, San Francisco, CA, 1993.
[3] W. Kent, A simple guide to five normal forms in relational database theory, Communications of the ACM 26 (1983) 120–125.
[4] K. Cios, G. Moore, Uniqueness of medical data mining, Artificial Intelligence in Medicine 26 (2002) 1–24.
[5] C.J. McDonald, The barriers to electronic medical record systems and how to overcome them, Journal of the American Medical Informatics Association 4 (1997) 213–221.
[6] S. Bergamaschi, S. Castano, M. Vincini, Semantic integration of semistructured and structured data sources, ACM SIGMOD Records 28 (1999) 54–59.
[7] R. Los, A. Ginneken, M. De Wilde, J. Der Lei, OpenSDE: row modeling applied to generic structured data entry, Journal of the American Medical Informatics Association 11 (2004) 162–165.
[8] J.C. Prather, D.F. Lobach, L.K. Goodwin, J.W. Hales, M.L. Hage, W.E. Hammond, Medical data mining: knowledge discovery in a clinical data warehouse, in: AMIA Annual Fall Symposium, 1997, pp. 101–105.
[9] A. Rector, W. Nolan, S. Kay, Foundations for an electronic medical record, Methods of Information in Medicine 30 (1991) 179–186.
[10] C. Strauch, U.L.S. Sites, W. Kriha, NoSQL Databases, Lecture Notes, Stuttgart Media University, 2011.
[11] R. Cattell, Scalable sql and nosql data stores, ACM SIGMOD Records 39 (2011) 12–27.
[12] H. Roitman, S. Yogev, Y. Tsimerman, D.W. Kim, Y. Mesika, Exploratory search over social-medical data, in: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, ACM, Glasgow, Scotland, UK, 2011, pp. 2513–2516.
[13] J. Yang, D. Tang, X. Zheng, Research on the distributed electronic medical records storage model, in: 2011 International Symposium on IT in Medicine and Education (ITME), 2011, pp. 288–292.
[14] C. Catley, M. Frize, A prototype XML-based implementation of an integrated 'intelligent' neonatal intensive care unit, in: 4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine, 2003, pp. 322–325.
[15] G. Stalidis, A. Prentzab, Medical support system for continuation of care based on XML web technology,

International Journal of Medical Informatics 64 (2001) 385–400.
[16] K. Araki, K. Ohashi, Medical markup language (MML) for XML-based hospital information interchange, Journal of Medical Systems 24 (2010) 195–211.
[17] H. Jagadish, S. Al-Khalifa, A. Chapma, TIMBER: a native XML database, The VLDB Journal 11 (2002) 274–291.
[18] W. Meier, eXist: an open source native XML database, in: A. Chaudhri, M. Jeckle, E. Rahm, R. Unland (Eds.), Web, Web-Services, and Database Systems, Springer, Berlin, Heidelberg, 2003, pp. 169–183.
[19] T. Fiebig, S. Helmer, C.-C. Kanne, G. Moerkotte, J. Neumann, R. Schiele, T. Westmann, Anatomy of a native XML base management system, The VLDB Journal 11 (2002) 292–314.
[20] G.I. Paterson, M. Shepherd, W. Xiaoli, C. Watters, D. Zitner, Using the XML-based clinical document architecture for exchange of structured discharge summaries, in: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS), 2002, pp. 1200–1209.
[21] S. Li, F. Chen, Native-XML storage method of electronic medical record, Computer Aided Engineering 4 (2006).
[22] I. de la Torre, F. Díaz, M. Antón, J. Díez, B. Sainz, M. López, R. Hornero, M. López, Choosing the most efficient database for a web-based system to store and exchange ophthalmologic health records, Journal of Medical Systems 35 (2011) 1455–1464.
[23] T. Beale, Archetypes: constraint-based domain models for future-proof information systems, in: OOPSLA 2002 Workshop on Behavioural Semantics, 2002, pp. 1–18.
[24] P. Schloeffel, T. Beale, G. Hayworth, S. Heard, H. Leslie, The relationship between CEN 13606, HL7, and openEHR, in: HIC 2006 Bridging the Digital Divide: Clinician, Consumer and Computer, 2006.
[25] K.W. Boone, K.W. Boone, The HL7 clinical document architecture, in: The CDA TM Book, Springer, London, 2011, pp. 17–21.
[26] N. Lavra, Selected techniques for data mining in medicine, Artificial Intelligence in Medicine 16 (1999) 3–23.
[27] G. Weglarz, Two worlds of data – unstructured and structured, DM Review 14 (2004) 19–22.
[28] R.K. Los, A.M. van Ginneken, J. van der Lei, Extracting data recorded with OpenSDE: possibilities and limitations, International Journal of Medical Informatics 74 (2005) 473–480.
[29] J. Golick, Introducing Friendly: NoSQL with MySQL in Ruby, Available from: http://jamesgolick.com/2009/12/16/introducing-friendly-nosql-with-mysql-in-ruby.html
[30] J.D. Ullman, J. Widom, A First Course in Database Systems, 3rd ed., Prentice Hall, Upper Saddle River, NJ, 2007.
[31] P. Walmsley, XQuery, O'Reilly Media, Sebastopol, CA, 2007.