



Removing the Redundancies in XML Document Design using GN-DTD

Ms.Jagruti S.Wankhade* and Prof. Vijay S. Gulhane

Department of I.T.

Sipna's college of Engg and Tech.

Amravati, Maharashtra, India.

Jagruti_wankhade22@rediffmail.com*

V_gulhane@rediffmail.com

Abstract- As XML becomes widely used, dealing with redundancies in XML data has become an increasingly important issue. Redundantly stored information can lead not just to a higher data storage cost, but also to increased costs for data transfer and data manipulation, such data redundancies can lead to potential update anomalies. One way to avoid data redundancies is to employ good schema design based on known functional dependencies. This paper presents a graphical approach to model XML documents based on a Data Type Documentation called Graphical Notations-Data Type Documentation (GN-DTD). GN-DTD allows us to capture syntax and semantic of XML documents in a simple way but precise. Using various notations, the important features of XML documents such as elements, attributes, relationship, hierarchical structure, cardinality, sequence and disjunction between elements or attribute are visualize clearly at the schema level.

Keywords- XML Model, GN-DTD design, Normalization Rules, Transformation Model

I. INTRODUCTION

With the wide exploitation of the web and the accessibility of a huge amount of electronic data, XML[1,14] (extensible Mark-up Language) has been used as a standard means of information representation and exchange over the web. Additionally, XML is currently used for many different types of applications which can be classified into two main categories [2,4]. The first application is called *document centric* XML and the other is called *data centric* XML. The document centric XML is used as a mark-up language for semi-structured [3] text documents with mixed-content elements and comments. The data centric XML consists of regular structure data for automated processing and there are little or no element with mixed content, comments, and processing instruction. The current XML data models however do not pay sufficient attention to the Problem of representing the structure of XML documents [4]. We believe, in order to present more sophisticated forms of XML documents structure, the schema such as DTD or XML schema must taken into account since it is used to define and validate XML documents structure. In our work, we consider DTD, as it has been widely well accepted and expressive enough for a large variety of applications. Furthermore DTD is an early standard for XML, and many legacy XML documents structures are defined by DTDs.

In this paper, we proposed a graphical notation of DTD called GN-DTD to overcome the above limitations. The GN-DTD[5] helps to arrange the content of XML documents in order to give a better understanding of DTD structures, to improve an XML design[9] and normalization[6] process as well. GN-DTD has richer syntax and structure which incorporate of attribute identity, simple data type, complex data type and relationship types between the elements. Furthermore, the semantic constraints that are important in XML documents are defined clearly and precisely to express the semantic expressiveness. We believe, GN-DTD can be used

to represent and support XML structure and capture more semantic of XML documents in order to solve some difficult issues, for example, query processing, lossless information and normalization.

II. RELATED WORK

Major current XML data models use directed edge labelled graphs to represent XML documents and their Schemas. These models consist of nodes and directed edges which respectively represent XML element in the document and relationship among the element. These existing XML model can be categorised into: XML model to represent instance of XML document, XML model represent XML schema and XML model for representing both XML document and XML schema. Examples are DOM [7] (document object model), OEM (object exchange model), S3-GRAPH and many more. As a summary, data models such as OEM, DOM, Data Guide have been designed for the purpose of information or schema integration. The focus of these data models is on modelling the nested structure of semi structured data but not modelling the constraint that hold in the data. In contrast, data model such as S3-Graph, CM Hyper graph, EER, XML Trees and ORA-SS [8] have been defined specifically for data management. Amongst these models, the notation of ORA-SS, semantic network model and EER notations are best to be adopted and applied in GN-DTD. Note that our notation is different from ORA-SS and Semantic Network since we have explicitly distinguished between complex element and simple element. We also made the ordering of sub element significant by treating them as a sequence.

III. XML MODEL DESIGN

Consider the DTD in Fig. 1 describes the XML Documents in Fig. 2. The first line of DTD in Fig. 1 shows that *department* is the root of the DTD. While second line shows that *department* consists of sub element *course*. The

semantic relationship between *department* and *course* is indicated by the symbol *, represents that *department* can consists of zero or many *course* for each department. The third line of the DTD shows that each element *course* has sub element *title* and element *taken by*. Symbol “,” between them indicated that they must occur in sequence. The fourth line indicates that element *course* has an attribute *cno*. The keyword ‘#REQUIRED’ represents that the attribute *cno* must appear in every *course* while “ID” indicates that the value of *cno* is unique within XML document. The fifth line of the DTD shows that the keyword “PCDATA” to despite that element *title* has no sub element and it is a leaf element and has a string value. The Fig.2 shows the XML document conforming to the rules stated in DTD in Fig. 1.

```
<!DOCTYPE department[
<!ELEMENT department(course*)>
<!ELEMENT course(title,taken_by)>
<!ATTLIST course cno ID #REQUIRED>
<!ELEMENT title(#PCDATA)>
<!ELEMENT taken_by(student*)>
<!ELEMENT student(firstname|lastname?,teacher)>
<!ATTLIST student Sno ID #REQUIRED>
<!ELEMENT title(#PCDATA)>
<!ELEMENT taken_by(student*)>
<!ELEMENT student(firstname|lastname?,teacher)>
<!ATTLIST student Sno ID #REQUIRED>
<!ELEMENT firstname(#PCDATA) >
<!ELEMENT lastname(#PCDATA) >
<!ELEMENT teacher (tname)>
<!ATTLIST teacher tno ID #REQUIRED>
<!ELEMENT tname (#PCDATA)
```

Figure 1:DTD Structure Design

ITS related XML document confirms to dtd is as follows

```
<!DOCTYPE courses [
<courses>
<course>
<course cno = “csc101”>
< title > XML database </title>
< taken_by>
< student >
<student sno = “112344”>
<firstname> zurinahni</firstname>
<lastname> zainol </lastname>
<teacher>
<teacher tno=“123”>
<tname>Bing </tname>
</teacher>
</student>
< student >
<student sno = “112345”>
<firstname> Azli </firname>
<teacher>
<teacher tno = “123”>
<tname> Bing </tname>
</teacher>
</student>
<course>
<course cno = “csc102”>
< title > Database Design </title>
<taken_by>
< student >
<student sno = “112344”>
<firstnme> zurinahni</firname>
<lastname>zainol </lastname>
<teacher>
```

```
<teacher tno = “123”>
<tname> Botaci </tname>
</teacher>
</student>
< student >
<student sno = “112345”>
<firstnme>Azli </firstname>
<teacher>
<teacher tno = “123”>
<tname> Botaci </tname>
</teacher>
</student>
</course>
</courses>
```

Figure 2: XML document related to above DTD

Any XML document [9,15] that satisfies and conforms to this DTD is likely to contain data redundancies which may lead to update anomalies. For example, as shown in Figure 2, the lecturer named *Bing* who teaches the same *course number* (*cno*) *csc101* is stored twice, which will lead to the updating anomalies. To avoid such problems, a set of rules should be provided when designing a DTD for XML documents.

The objective of this work is to provide a methodology which simplifies the process of designing a non redundancy XML document[10,16]. To achieve this, a conceptual model called GN-DTD is proposed. GN-DTD is a graphical modelling approach for describing both DTD and XML documents. For GN-DTD itself, we define a complete set of syntax and structure which incorporates attributes, simple data type, complex data type, and types of relationship among them. Furthermore, semantic constraints are also precisely defined in order to capture semantic meaning among those defined objects. In this work, we present normal forms for GN-DTD based on both Arenas and Libkin's rules [1] and Ling et al's rules [11,18] in a simpler form to allow users/designers to find an 'optimal' structure of XML elements/attributes[12,14]. This will produce a correct, complete and consistent representation of the real world XML data which may benefit the users. We ensure that DTD mapped from GN-DTD are similar to XNF [13].

IV. TRANSFORMATION OF DTD INTO GN-DTD

GN-DTD emphasizes the representation of semantic constraints between the complex elements, simple elements and attributes clearly. GN-DTD represents the structure and the semantic constraints of the XML document in a schema level. GN-DTD has following basic components:

- a. A set of complex element node representing the element that have sub element
- b. A set of simple element nodes representing simple element that have no sub element
- c. A set of attributes nodes representing the attributes defines in ATTLIST.
- d. A semantic relationship between two nodes.
- e. A root node

```
Consider following DTD
<! DOCTYPE department [
<! ELEMENT department (course*)>
<! ELEMENT course (title, student*)>
<! ATTLIST course cno ID #REQUIRED>
<! ELEMENT title (#PCDATA)>
<! ELEMENT student (fname|lname?,lecturer)>
```

```
<! ATTLIST student Sno ID #REQUIRED
<! ELEMENT fname(#PCDATA)>
<! ELEMENT lname(#PCDATA)>
<! ELEMENT lecturer(tname)>
<! ATTLIST lecturer tno ID #REQUIRED>
<! ELEMENT tname (#PCDATA)>
]>
```

Figure 3: DTD

Following is the list of some notations used by GN-DTD model to represent DTD into Graphical form

Notation	Meaning
	Complex element
	Mandatory simple element, single value, CDATA
	Required simple element, multi value, CDATA
	Optional simple element, single value, CDATA
	Optional simple element, multi value, CDATA
	Composite Attribute
	Reference Attribute
	Required Attribute
	Reference attribute
Notation	Meaning
	Part-of link simple element (complex element and simple element)
	Part-of link attribute (Complex element and attribute)
	n-ary one-to-many inheritance relationship
	n-ary many-to-many inheritance relationship
	n-ary many-to-one inheritance relationship
	Sequence between set of relationship
	Disjunction between set of relationship

A. Constraint Between Set Of Relationship:

a. Sequence between Set of Child Element Nodes:

Normally each complex element node consist a single attribute node or multi attribute node. We emphasize in our notation those node must be located first in the sequence before include other simple or complex elements node. To

illustrate this, we draw a directed curved up arrow and labelled with {sequence} across all the set of relationship involved. Consider the following segment of DTD and its GN-DTD where attribute *Sno* is located at first position in the sequence of child elements.

```
<! ELEMENT student (fname,lname,grade)>
<! ATTLIST student Sno ID #REQUIRED>
<! ELEMENT fname(#PCDATA) >
<! ELEMENT lname(#PCDATA) >
<! ELEMENT grade(#PCDATA) >
```

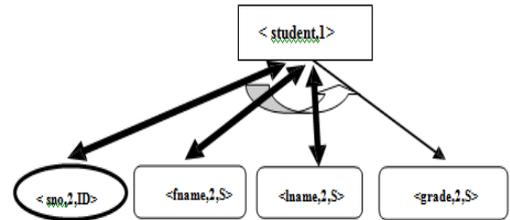


Figure 4: Sequence of Attributes

b. Sequence Between The Set Of Sub Element:

We have a set of sub elements that are in an exclusive “OR” {XOR} relationship to represent notation “[in DTD. For example, for the complex element node *student*, only one of its sub elements which are *fname* or *lname*, to be appeared as its sub elements in the XML document. To illustrate this, we draw a line and labelled with {XOR} across all the set of relationship involved. Follows is a real example of application.

```
<! ELEMENT chapter (page| citation| table)* >
which is equivalent with
<! ELEMENT chapter (page*| citation*| table*) >.
```

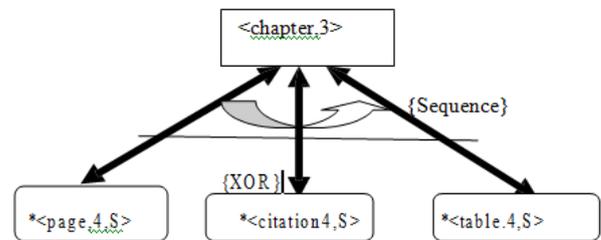


Figure 5: Disjunction of several Simple Element

Following is GN-DTD form of Fig.3 DTD

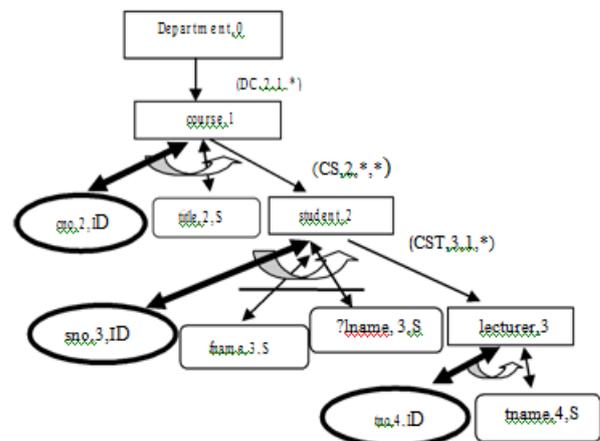


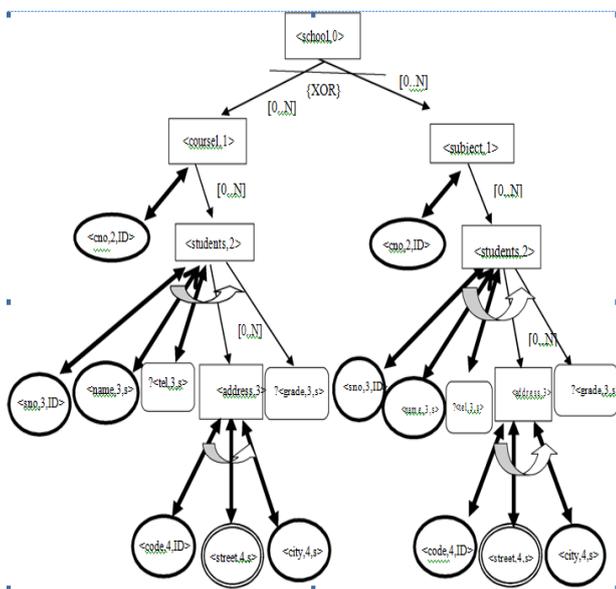
Figure 6: GN-DTD Representation

TO better understand, consider the following DTD

```

<! DOCTYPE school[
<! ELEMENT school (course*|subject*)>
<! ELEMENT course(students*)>
<! ATTLIST course cno ID #REQUIRED>
<! ELEMENT subject(students*)>
<! ATTLIST subject sno ID #REQUIRED>
<! ELEMENT students (student*)>
<! ELEMENT student ( tel?, address*,grade?)>
<! ATTLIST student Sno ID #REQUIRED>
Name CDATA #REQUIRED>
<! ELEMENT tel (#PCDATA)>
<! ELEMENT address (EMPTY)>
<ATTLIST address Code (CDATA)
#REQUIRED street (CDATA) #IMPLIED
City (CDATA)#REQUIRED>
<! ELEMENT grade (#PCDATA)>

```



This is The main Diagrammatical Representation of DTD on which we are going to apply the Normalization Rules to delete all the redundancies, anomalies which makes the XML as a bad XML document.

V. NORMALIZATION RULES FOR GN-DTD

A. First Normal Form GN-DTD(1XNF GN-DTD):

The first normal form [1, 13] for GN-DTD is about finding unique identifier attributes for the complex elements set, and checking that no node (complex element, simple element or attribute) actually represents multiple values. To be in first normal form, each attribute, complex element or simple element is not NULL and has a single label. More importantly, the primary key (unique identifier) for the complex element must be defined.

- a) Only one value for each simple element node or attribute node of GN-DTD can be stored. If there is more than one value, we must add some new element nodes or attribute nodes to store them.
- b) The root element of a GN-DTD model should be located at level 0 and the cardinality of the root element node must be one.
- c) Each set of complex element node in the GN-DTD has

at least one key attribute node.

B. Second normal form (2XNF GN_DTD):

Some nodes need to be restructured. However they can then still be in a single GN-DTD. This is possible in XML because XML supports hierarchies in a single document, while relational databases do not support hierarchies in a single row. This is different from the relational second normal form (2NF), which requires one-to-many relationships to be in separate tables. The GN-DTD is in second normal form[1] if and only if:

- a) GN - DTD is in 1XNF.
- b) There is no nested binary inheritance relationship or ternary inheritance relationship under many-to-many or one -to-much inheritance relationships with the following condition: For each nested set of complex element <CE,l+1> of <CE,l>, and any key attribute (ATT) of <CE,l>, the key attribute and simple element of <CE,l+1> is not partial dependent on ATT of complex element <CE,l>

C. Third normal form (3XNF GN_DTD):

In the third normal form of the GN-DTD, making changes to one unique complex element node set would not affect the integrity of another complex element node sets .If needed; complex element node set would be divided into two separate complex element node set. GN-DTD is in third normal form if and only if:

- a) GN-DTD is in 2XNF.
- b) There exists no nested inheritance relationship type of n-ary many-to-one or many-to-many under a one-to-many inheritance relationship set in GN-DTD and the following conditions are satisfied:
 - i. For each nested set of complex elements <CE_b,l+1> of set of complex element <CE_a,l>, any key attribute and simple element of <CE_b,l+1> is not transitively dependent on ATT of complex element <CE_a,l>
 - ii. Any key attribute node of any complex element node located in a different level are disjoint (ATT<CE,l> ∩ ATT<CE,l+1> ∩ ATT<CE,n> =0)

D. Normal form GN-DTD(NF GN-DTD):

GN- DTD is in Normal Form[1,13] if and only if:

- a) GN-DTD is in 3NF.
- b) There are no global dependencies between attribute and simple element of complex element nodes under nested one-to-many or many-to-many inheritance relationship.

VI. TRANSFORMATION FROM GN-DTD TO DTD

After removing all the types of redundancies GN-DTD can be transform back to DTD structures Following is the set of some transformation rules used to come back to the original DTD

Step 1: Level 0, a root node is represented By <! DOCTYPE root node name [element type definition] >

Step 2: Level 1, identity the sub tree of GN-DT check the number of nodes, type of nodes and relationship type

Step 3: If there is no more than one node at level 1 and nodes are hierarchical then generate

<! ELEMENT root node name (Ni) >

Where Ni is the list of sub elements/child nodes

3.1 Check the relationship set between *parent* Nodes and *child* nodes,

3.1.1 If {XOR} means the relationship between node is a disjunction and will be represented using symbol ‘|’

Else

3.1.2 If {sequence} means the relationship is sequence and will be represented using symbol ‘;’

3.2 Check the semantic constraint between *parent* nodes and *child* nodes in each of relationship set and map to following operator:

3.2.1if [0..N] map to operator *,

3.2.2if [1..N] map to operator +

3.2.3if [0..1] map to operator ?

Step 4: If the list of sub elements (*Ni*) is not empty, using depth first traversal, for each node in list sub element *Ni*

4.1 repeat step 3.1 and 3.2

4.2 generate < ! ELEMENT *Ni* (sub element *Nj*)>

4.3 for each complex element (*Ni*), find an attribute node and generate

<! ATTLIST *Ni* attribute name attribute type>

4.4 For sub element *Nj*

4.4.1If *Nj* is a simple element has part of link with *Ni* then generate

<! ELEMENT simpleelement name #PCDATA>

(Repeat for all simple element nodes)

4.4.2 If *Nj* is a complex element node has inheritance link with *Ni*

Repeat step 4

4.4.3 If *Nj* is a complex element node has part of link then generate

<! ELEMENT *Nj* (EMPTY) >

Step 5 : Go to next sub tree GN-DTD and repeat step 4

VII. CONCLUSION

We have proposed a method for designing a “good” XML document in two steps: first, we building a conceptual model by means of GN-DTD at the schema level and second, using normalization theory where functional dependencies are refined among its simple elements and attributes. The GN-DTD can be further normalised either to 1XNF, 2XNF, 3XNF or XNF using the proposed normalization algorithm. In the proposed methodology, a GN-DTD is used as input and the normalization rules are applied during the normalization process. We also explain the process for transforming GN-DTD into DTD

VIII. REFERENCES

- [1]. Arenas, M. and Libkin, L. A Normal Form for XML Documents, ACM Transaction on Database System, vol 29(1), 2004, pp. 195-232.
- [2]. Vincet, M., Liu, J., Mohania, M., On the equivalence between FDs in XML and FDs in relations. Acta Infomatica, 2007, pp. 230-247.
- [3]. Ling, T.W., Lee, M.L. and Dobbie, G. Semi structured Database Design, Springer 2005.
- [4]. Wang, J. and Topor, R., Removing XML Data Redundancies Using Functional and Equality- Generating Dependencies, 16th Australasian Database Conference, 2005, pp. 65-74.
- [5]. Zainol, Z. and Wang, B., GN-DTD: Graphical Notation for Describing XML Documents, In Proceeding of 2nd International Conference on Advances in Databases, Knowledge, and Data Applications, DBKDA, IEEE

- Computer Society, 2010, pp. 214-221.
- [6]. Ling, T.W, A normal form for entity-relationship diagram, Proceeding 4th International Conference on Entity Relationship Approach, 1985, pp. 24-35
- [7]. Hegaret, P.L, The W3C Document Object Model (DOM), availableat [Http://www.w3.org/2002/07/26/domarticle](http://www.w3.org/2002/07/26/domarticle), 2002.[accessed January 10, 2009]
- [8]. Dobbie, G., Xiaoying, W., Ling, T.W., and Lee, M.L. ORA-SS: An Object-Relationship- Attribute Model for Semi-Structured Data. Technical Report, Department of Computer Science, National University of Singapore, 2001.
- [9]. Embley, D. and Mok, W.Y., Developing XML Documents with guaranteed "good" properties, In Preceedings of the 20th International Conference on Conceptual Modeling, 2001, pp. 426-441.
- [10]. Fallside, D.C. XML Schema Part 0; Primer, W3C Recommendation, available at <http://www.w3.org/TR/XMLschema-0.2001>. [accessed November 20, 2009]
- [11]. Lee, S.Y., Lee, M.L., Ling, T.W., and Kalinichenko, L.A., Designing Good Semi-structured Databases, 1999, pp. 131-135.
- [12]. Goldman, R., and Widom, J., Dataguides: Enabling query formulation and optimization in semi structured database, In Proceeding of the 23rd International Conference on Very Large Databases (Athens), 1997, 436-445.
- [13]. Lv, T., Gu, N., Yan, P., Normal forms for XML documents, Information and Software Technology, 2004, pp. 839-846.
- [14]. Powell, G., Beginning XML Database, USA: Wiley, 2007
- [15]. Bex, G. J., Neven, F., and Bussche, J.V., DTD versus XML Schema. A Practical Study. In Proceeding of the Seventh International Workshop on the Web and Databases, 2004, pp. 79-84.
- [16]. Yu, C. and Jagadish, J.H., XML schema refinement through redundancy detection and normalization, The VLDB Journal, 2008, pp. 203-223.
- [17]. Feng, L., Chang, E., and Dillon, T., A Semantic Network-Based Design Methodology for XML Documents, ACM Transactions on Information Systems, Vol 20, Number 4, 2002, pp. 390-421.
- [18]. Ling, T.W and Yan, L.L, NF-NR: A practical Normal Form for Nested Relations, Journal of System Integration, 4, 1994, pp. 303-340
- [19]. Chen, P.P. The entity-relational model: Towards a unified view of data, ACM transaction on Database System, 14, 1976.
- [20]. Biskup, J., Achievements of relational database schema design theory revisited, Semantic in Database, LNCS, vol 1066, Springer, 1995, pp14-44.
- [21]. Gustas, R., A Look Behind Conceptual Modelling Constructs in Information System Analysis and Design, International Journal of Information System Modelling and Design, 1(1), 2010, pp.79-108
- [22]. Mani, M., Lee, D., and Muntaz, R.R., Semantic Data Modeling Using XML Schemas, In Proceeding of 20th International Conference on Conceptual Modelling. 2001
- [23]. Mok, W. (2002). A comparative Study of Various Normal forms. IEEE Transactin on Knowledge and Data Enginnering, Vol.14 , pp. 369-385.
- [24]. Mok, W.Y., Ng, Y.K and Embley, D.W., A normal for precisely characterizing redundancy in nested relation, ACM Transaction Database System, Vol. 12, no.1, 1996. pp. 77-106
- [25]. Ozsoyoglu, Z.M and Yuan, L., A new normal form for Nested Relations, ACM Transaction on Database

- System, Vol. 12, No 1, 1987, pp.111-136.
- [26]. Papakonstantinou, Y., Garcia-Molina, H. and Widom, J. Object exchange across heterogeneous information sources. Proceedings of the Eleventh International Conference on Data Engineering: 1995, pp. 251–260.
- [27]. Halpin, T., Object Role Modelling: Principle and Benefit, International Journal of Information System Modelling and Design, 1(1), 2010, pp. 33-55.
- [28]. Codd, E. Further Normalization of the Data Base Relational Model. In Database system, Computer Science Symposia series 6. Prentice Hall,1972 [29]W3C XML Specification DTD, available at [Http://www.w3.org/XML/1998/06/xmlspecreport19919010.htm](http://www.w3.org/XML/1998/06/xmlspecreport19919010.htm) [accessed December 29, 2009]
- [29]. Bird,L., Goodchild, A., and Halpin, A. Object Role Modeling and XML-Schema. ER2002, pp.309-322.
- [30]. Kolahi, S., Dependency-preserving normalization of relational and XML data, Journal of Computer and system Sciences, 2007, pp. 636-647.
- [31]. Yuliana, O.Y. & Chittayasothorn, S. (2005). XML Schema Re-Engineering Using a Conceptual Schema Approach. Internatinal Conference on Information Technology: Coding and Computing. IEEE.
- [32]. Buneman, P., Fan, W., Simeon, J., and Wienstein, S. Constraints for Semi structured data and XML. SIGMOD record 30, 2001, pp. 47-54.