Data Integration with XML ETL Processing

G.Jayashree
Research Scholar,
Department of Computer Science,
VELS Institute of Science, Technology and Advanced Studies
Chennai, India.
Studies,
Jayashreegopalsamy@gmail.com

Dr.C.Priya,
Research Supervisor,
Department of Information Technology,
School of Computer Science,
VELS Institute of Science, Technology and Advanced

Chennai, India.
drcpriya.research@gmail.com

***Abstract*---**Data forms the key success enabler for any business organization. Structured collection and organized data along with this analytical enablement will provide more insights to the organization's existing operations and ways for better improvement. In order to do efficient data collection, integration of data from various sources forms the vital role. Many of the vital business initiatives for an organization requires the data to integrate. Such initiatives performed by creating a customized solution across every data source. In such given architecture, the complexity to integrate data source is proportionate to the increase in data sources. Use of eXtensible Markup Language (XML) for standardizing the information exchange across several sources is an industry standard implementation. It is the technology used to implement and deliver an enterprise-level data integration solution using heterogeneous data sources. The transformation capability of XML is capable of handling very complex and big documents and can do processing of ~thousands of real-time transactions using message bus or Web Services methods. The XML provided solutions are flexible and easily adaptable, thus enabling them to incorporate into a broader range of organizational solutions. This paper discusses in detail, the data integration methods and their challenges and processing of XML data using Extraction, Transformation and Load method. The paper also provides the several ways to parse the XML data as part of the data processing with the case study for an On Line Transaction Process (OLTP) system of a manufacturing company.

Keywords: XML, Data Integration, ETL, Data Virtualization, XML Parsing, OLTP

I. INTRODUCTION

Data integration (DI) is an iterative process consisting of data merging from several heterogeneous source systems, and stored with the help of several technologies thus providing a unified glimpse of data. It is gaining its importance where we require the merge or integrate systems of any two companies or requiring application consolidations within one company to provide a single view of the company's assets. The later initiative is the data warehouse. Merging of diverse data types (tables, data sets and documents, etc.) by any application, user or by an organization is possible in data integration. DI does the alignment, combine and presentation of every dataset of the internal departments and remote sources of larger data sets and hence by providing the support for analytical data processing. The spreadsheet consolidation/merging in a document in Microsoft word is a simple example for DI in a small paradigm.

A. XML – A Glimpse

XML defined as Extensible Markup Language. It is a medium for data language exchange, which is independent on platforms. Though referred as a data language, XML can also store information. It was create to describe the data, its content and structure. An XML file possesses the below general characteristics:

- Uses the character set
- Uses the tags (for increasing the complexity with 90%of the complete size of the XML file)
- Having a Complex structure
- Uses the DTD (Document Type Definition) and XML schema to identify the structure and grammar of documents.

Disparate group of people can provide the XML DTD's for in-between data exchanges. It can also use to validate the document propriety. Structure of DTD, is designed to be complicated in nature and it contains: structures for base data elements mixed content, cardinality, nullable and other allowed values. The other standard used to define the structure for XML documents is the XML schema, which is more effective than DTD. It allows doing the following:

- Defining the elements for documents along with their required attributes
- Defining the elements' sequence and hierarchy
- Defining the elements' values (empty or has any text)
- Defining extensibility for future additions
- Namespace Support (To identify whether the definition for the

elements or attributes can be obtained or not)

B. *Advantages of XML*

XML, with its self-describing data format, allows the storage of diverse data (with/without an XML schema) in a single row or document, and simultaneously providing the capability to aggregate or search the data. Applications will be able to evolve their XML schemas with no changes to the underlying database schema. Some of the other key advantages of XML are as follows:

- Easily understandable and readable,
- Easier to code when compared with HTML coding
- Compatible with Java™ and completely portable. Applications processing XML data can use the information across any platform.
- Extendable. Able to create individual tags or can use the tags created by others.
- Easily understandable by the interfacing system.
- Capable of handling Meta data and large data
- Ability to handle complex structures.
- Use of XML structure (XSD) to import XML source files. It is widely used in integrating with standard systems like ECC ERP, which has a standard and defined XSD structure data flow.
- XML structures has capabilities to handle big data set and other fields structure

C. *Applications of XML*

XML has widely been used for e-business, portable and Web applications.

Web publishing: With XML, we can create interactive pages, allowing the page customization to the customers, and creating intuitive e-commerce applications. Data can be stored once and retrieved many times for different devices or viewers or depending on the style-sheet processed using the Extensible Style Language (XSL)/XSL Transformation (XSLT) processor

Automation of Web tasks and web surfing: XML clearly defines the information type required in a document, hence-by making easier to render meaningful results while searching the Web: For example, fpr searching the books authored by "Tom Brown" using HTML can possibly return the instances of the word 'brown' outside the author context. XML restricts the correct search context here (the information present in the <author> tag) and returns only the required information. Web-based Robots (Program for Web search automation) and Web agents can provide more efficient and meaningful results by using the XML.

General applications: A standard method provided by XML to access data, which makes easier for devices and applications to store, use, transmit, and display data.

•**E-business applications**: Electronic data interchange (EDI) are widely accessible for data interchange, business-to-consumer transactions and business-to-business transactions, using XML implementations.
•Metadata applications: With the use of XML metadata can be represented in a portable and reusable format.

•**Pervasive computing**: XML renders structured and portable data for display wireless (pervasive) computing devices like cellular phones, personal digital assistants (PDAs), etc. Wireless Markup Language (WML) and VoiceXML are the current evolving standards for visual and speech-driven wireless device interfaces.

II. SCHOLARLY REVISIT

Elisa Bertino and Elena Ferrari, discussed in detail the benefits of XML being used a standard for data representation. The challenges related to the XML usage like formal foundation for web standards, support for metadata and invention of semantic-based tools for knowledge discovery explained in detail.

Rashed Salem, Jérôme Darmont & Omar Boussaid proposed a common metadata-driven and service oriented event driven approach to integrate the timely web data using Active XML (AXML).An incremental XML-based algorithm was proposed with associated mining rules for automating and reactivation tasks.

Lucas Zamboulis and Alexandra Poulovassilis proposed a data-integration framework with two specific algorithms for XML data integration. The first algorithm is for the integration of schema and the second algorithm is for view materialization, both of them based on the graph restructuring.

Chong-Shan Ran and Ma-Chuan Wang proposed the data integration based on XML schema mappings. In this approach, the relation between global and local schemas represented with XML

schemas. In addition, the entertaining manager present in the schema allows managing the data models separately. An example focused on the relationship and role of both local and global schema and mapping of tables proposed as part of the implementation.

Yue Guohua and Wang Jingting provided the new approach to extract and load semi structured XML data into data warehouse. With an example of Book Return Data (BokeDataInfo.xml), the characteristics of the semi-structured content was analyzed with the DOM (Documents Object Model) parser and extracted the semi-structured XML data into Data warehouse ETL and loaded into the Data warehouse. The paper also addresses the disadvantages with the commercial ETL tools in loading the Semi structured data.

Nawfal El Moukhi, Ikram El Azami and **Abdelaaziz Mouloudi** proposed the data integration framework, which integrates the data from relational databases and developed the approach to design XML based data warehouse with the Model Driven Architecture (MDA) techniques. A list of rules to identify the data warehouse components from the relational data has been defined in the paper. The given rules will form the core components of the transformation engine of the data model with which the relational model transformed into the multidimensional model with MDA strategy.

III. DATA INTEGRATION METHODS AND CHALLENGES AND XML SUPPORT

A. Methods of Data Integration

Several integration methodologies or approaches that exist currently for implementing a data integration solution. They are

Data Consolidation – Consolidates data from many external systems and creates a single version of the consolidated data in one data store. ETL (Extract, transform, and load) technology supports for data consolidation. It pulls source data, applies the transformations and make it to use in an understandable format, and transfers to another database or data warehouse.

Data Propagation – Used to copy application data from any location to the other location. This is primary event-driven and can be designed either in a synchronous/ asynchronous way. Synchronous method of data propagation can support a bi-way exchange of data against source and target. Enterprise data replication (EDR) and Enterprise application integration (EAI) technologies can support data propagation.
EAI – Integration of system applications for the message and transaction exchange. It is often used for real-time business transaction processing.
EDR -- Larger volume of data transfer across databases. Logs and triggers captures and disseminates the changes in data.

Data Virtualization – An interface provides the consolidated data view in near real-time using various data models and different data sources**.** The data view and data storage are situated in different locations and not in a single location. Virtualization does the data retrieval and interpretation, but need not require any uniform format or any single point of access.

Data Federation -- A technical form of virtualizing data and used as a virtual database. A common data model is created and viewing of data from a single point of access. Enterprise information integration (EII) technology supports data federation. Data abstraction is used to perform a consolidated data view.

Data Warehousing -- Storage repositories for data. Performs the cleansing, re-formatting data for storage, nothing but the data integration.

B. Challenges in Data Integration

Data integration (DI) process is a tedious activity requiring data to get reconcile at different levels like data instance, data schema and data model. The need for a strong solution for data organizing into a common syntax plays a vital role here and XML founds to be the best fit for this Given below are some of the integration challenges.

Data model – Data in external sources are with different structures like tables, objects, and files to represent data. The heterogeneity model of data indicates the necessity for a generic data model that maps the in-coming data from several sources.

Data Schema – An entity or property in a data model can be represent differently within a schema. Also, data sources can represent the same information with variant data structures. The automation solution should address the given

differences while performing the data integration.

Data Instances – Possible integration issues at instance-level includes the determination of whether the distinct objects from disparate sources represents the same entity and identification of a single data source if any contradictory data is present in data sources, E.g. A person with varied birth dates.

C. XML to overcome Data Integration Challenges

1. XML organizes the data based on graph-based and hierarchical representations and facilitates the representation of structured, semi-structured, and unstructured information. It provides a flexible, solid and easy-to-manage **common data model** across the Web.

2. XML support extensibility, which means both the name and meaning of the XML tags are arbitrary. Hence, there has to be a standard domain-specific tags and schemas for integrating of data. In addition, the data integration solution should have an algorithm that describes the semantics to understand the data attributes and elements. E.g., A data element called "money" comes from two data sources with different currencies, mandates that the semantics of the data element to be investigated in detail before providing the solution. This brought into the ideology of metadata collection and any metadata be expressed using XML itself.

The process of **schema-level reconcile** provides the details about context, like meanings for a given name or any specific value which depends on the content where the information can occur. Hence, the interchanging and integrating of contextual information becomes the critical for any data integration technique/tool to implement.

IV. XML ETL PROCESSING

A. XML Data Processing methods

XML defined to be user-defined, platform independent and text based which can be easily understandable.

The most common and well-known version of XML is HTML (markup language designed for web). In the data integration world, XML standard primarily applied for communication across applications. Majority of the commercial ETL tools in market have their own dedicated XML components to handle the processing. Every ETL process for the XML processing task has its challenges and needs techniques to overcome it.

Some ETL tools provides several approaches to process and do transformation on XML data. Majority of the XML based ETL processing tasks likes to adopt to any one of the below given ways in which data interpreted and represented:

Event-based XML model:- A unique way of interpreting a series of events as XML. Every data string of XML, which is coming, represented as a separate entity – called an event. In this approach, all the events converted in the data flow as a record, with a dedicated filed used to store every attribute and tag. This model can work with all kinds of XML documents with the flexibility to adjust elements as required. This model will not provide the similar level of support for XML Schema, at some instances the model ignores several portions of the document resulting in incomplete data. It is an easy way to interpret XML strings.

Full-XML parsing model:– While reading a XML document, the ETL tool component which is being used, is aware of the proper structure of the document and parses it as appropriate. This needs additional effort to set up during the preliminary phase, but can increase the performance and makes sure the proper field mapping of all required values and empty fields. The other outcome of this approach is a reduction in time to populate the downstream components, as it is a more error proof. This model representation is the useful choice and most powerful, while working on complex XML schemas.

B. Data Processing – Case Study

A hypothetical manufacturing company is using two operational systems:

An On Line Transaction Processing (OLTP) system, for managing the daily operational data. This includes the daily details about orders, customers, invoices, products, etc. Data entered manually by the customer service executives.

A web-based application in which, customers place their purchase orders. Currently no integration between the given systems. When a user places an order, the details captured in an XML file, which is present in the web server. For every new order, a separate XML file is generated which has the data present in the internet form.

The orders are stored in a separate directory of the web server and the entire collection processed on a daily basis (overnight). The requirement is to load

the purchase orders data from the multiple XML files into an operational table.

Proposed solution

The high-level solution data flow for the given scenario depicted below in Fig 4.1.

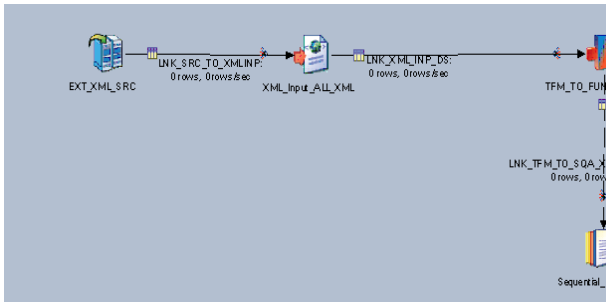

Fig. 4.1 Data flow for XML Processing

Given below are the steps to follow for any XML ETL Processing:

1. Movement of XML source files to ETL server from web server (to the *unprocessed_files* folder)
2. For every individual XML files,
   a. Parsing to be done on XML file based on event XML approach
   b. Additional data to get passed through the processing flow. E.g. file name containing dates and timestamps
   c. Validation of records (E.g. check whether a customer exists in the system and the data is correct is or not)
   d. Assignment of order numbers for every purchase order
   e. OLTP table to get fed with open orders
   f. OLTP table to get fed with load history with a corresponding status
3. Movement the file to the *processed_files* folder after the completion of XML processing.

C. Implementing Data Parsing Approaches for XML Data

For parsing the XML input file, we can do it in the below two different ways.

Approach 1: <u>Reading from source in a single column</u>

In this approach, the entire source data extracted as a single column in the ETL Transformation editor and taken for processing. The below steps needs to be taken care for the implementation that needs to be applied into the Transformation Editor as given in the figures Fig 4.2, Fig 4.3 and Fig. 4.4.

1. Select the Source Program type as "Specific Program (s)"
2. Add Source Program
3. Enter the final delimiter string
4. Define the column name as "Filename"

STEP-1, 2

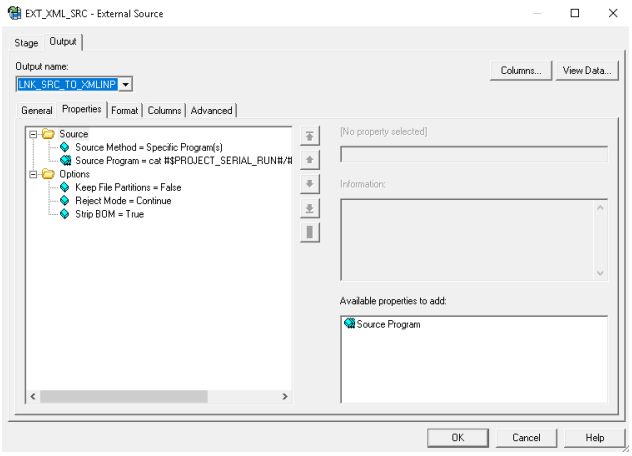

Fig 4.2 Program type selection

STEP-3

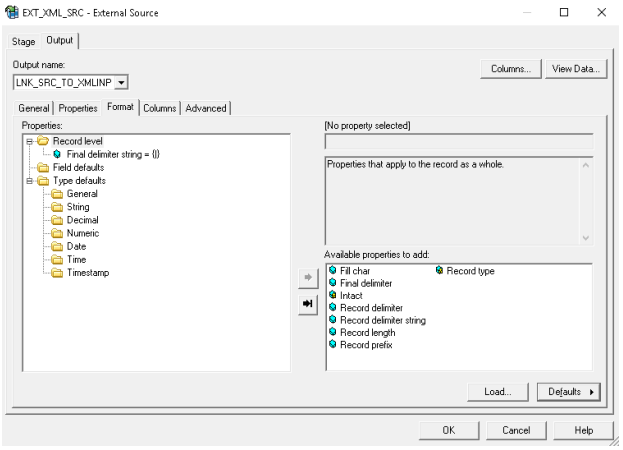

Fig 4.3 Delimiter String Selection

STEP-4

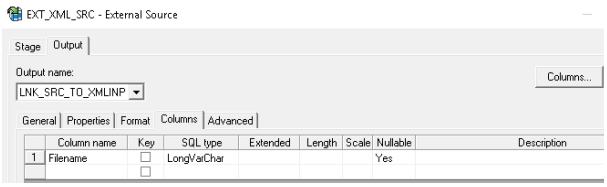

Fig 4.4 Definition of Column name

SAMPLE DATA

<xml>
<VerificationServiceResponse> <VerificationId>5800000000</VerificationId> <HasEligibleVerifier>true</HasEligibleVerifier> <VerifierList> <Verifier><VerifyType>PHONE</VerifyType><InEligible>false</InEligible><PhoneList><Phone> <CountryCode>1</CountryCode> <AreaCode>480</AreaCode><Number>8200000</Number><Voice>false</Voice> <InEligible>true</InEligible><InEligibleReason>RISK_RESTRICT</InEligibleReason><RiskScore>0</RiskScore><fullPhone>14808200000</fullPhone> </Phone> <Phone> <CountryCode>1</CountryCode><AreaCode>480</AreaCode><Number>7300000</Number><Voice>false</Voice><InEligible>true</InEligible><InEligibleReason>RISK_RESTRICT</InEligibleReason><RiskScore>0</RiskScore> <fullPhone>14807300000</fullPhone></Phone> <Phone><CountryCode>1</CountryCode> <AreaCode>480</AreaCode><Number>2000000</Number> <Voice>false</Voice><Text>true</Text><InEligible>false</InEligible> <RiskScore>0</RiskScore><fullPhone>14802000000</fullPhone></Phone></PhoneList></Verifier></VerifierList></VerificationServiceResponse><ValidatePhoneId>10000000</ValidatePhoneId></xml>{|}

RESULT

Filename = 5800000000 true PHONE false 1 480 8200000 false true RISK_RESTRICT 0 14808200000 1 480 7300000 false true RISK_RESTRICT 0 14807300000 1 480 2000000 false true false 0 14802000000

APPROACH 2: Reading XML content from single column and deriving multiple row

In this approach, the source data will be extracted and split into multiple columns in the ETL Transformation editor and will be taken for processing. The below steps needs to be taken care for the implementation that needs to be applied into the Transformation Editor as defined in Fig 4.5, Fig 4.6, Fig. 4.7 and Fig. 4.8.

1. Enable “grammar caching” in the Stage tab
2. Select the XML source column as “Filename” and column content as “XML document” in the Input tab
3. Enable the option “Inherit stage properties”
4. Define the column names in the Column tab

STEP-1

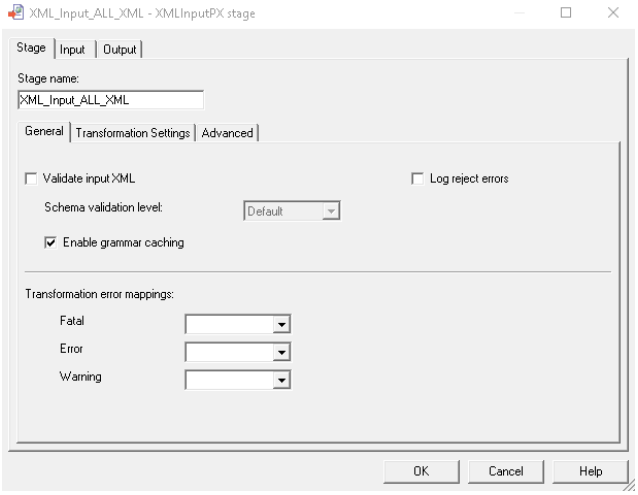

Fig 4.5 Grammar Caching Enablement

STEP-2

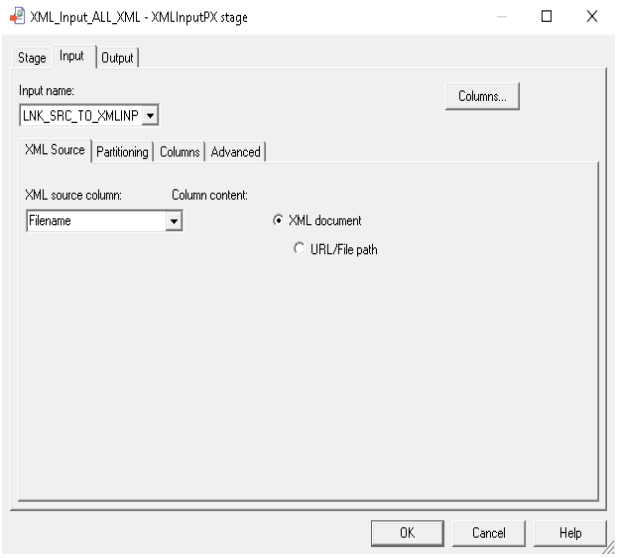

Fig 4.6 Selection of Column name

STEP-3

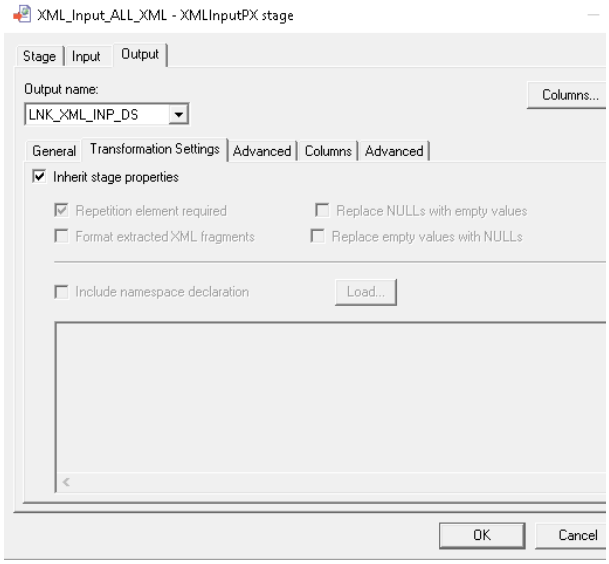

Fig 4.7 Inherit ion of Stage Properties

STEP-4

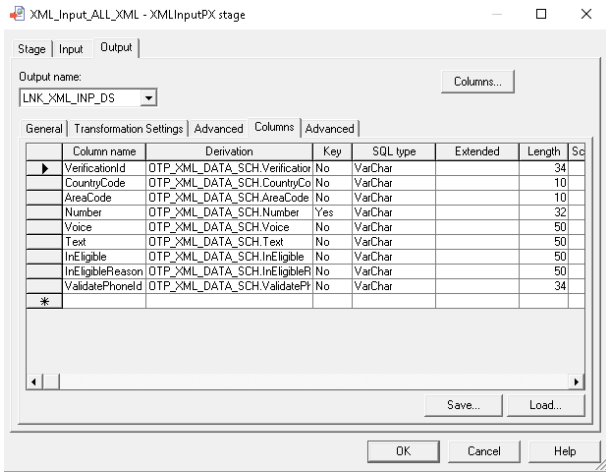

Fig 4.8 Definition of Column name

SAMPLE DATA

<xml>
<VerificationServiceResponse><VerificationId>5800000001</VerificationId><HasEligibleVerifier>true</HasEligibleVerifier><VerifierList><Verifier><VerifyType>DSID</VerifyType><InEligible>true</InEligible><InEligibleReason>DSID_NO_DATA_FOUND</InEligibleReason></Verifier><Verifier><VerifyType>SEC_QUESTIONS</VerifyType><InEligible>true</InEligible><InEligibleReason>SEC_QUESTIONS_NOTENROLLED</InEligibleReason></Verifier><Verifier><VerifyType>PHONE</VerifyType><InEligible>false</InEligible><PhoneList><Phone><CountryCode>90</CountryCode><AreaCode>216</AreaCode><Number>5700000</Number><Voice>false</Voice><InEligible>true</InEligible><InEligibleReason>RISK_RESTRICT</InEligibleReason><RiskScore>0</RiskScore><fullPhone>902165700000</fullPhone></Phone><Phone><CountryCode>90</CountryCode><AreaCode>549</AreaCode><Number>2700000</Number><Voice>false</Voice><Text>true</Text><InEligible>false</InEligible><RiskScore>0</RiskScore><fullPhone>905492700000</fullPhone></Phone></PhoneList></Verifier><Verifier><VerifyType>EMPID</VerifyType><InEligible>false</InEligible></Verifier></VerifierList>
</VerificationServiceResponse>
<ValidatePhoneId>9900000</ValidatePhoneId>
</xml>{|}

RESULT

VerificationId – 5800000001
HasEligibleVerifier – true VerifyType - DSID
InEligible – true CountryCode - 1
AreaCode – 480 Number - 8200000
Voice – false InEligible - true
InEligibleReason - DSID_NO_DATA_FOUND
RiskScore – 0 fullPhone – 14808200000

CountryCode – 1 AreaCode - 480
Number – 7300000 Voice - false
InEligible – true InEligibleReason -
RISK_RESTRICT

RiskScore – 0 fullPhone - 14807300000
CountryCode – 1 AreaCode - 480
Number – 2000000 Voice - false
Text – true InEligible - false
RiskScore – 0 fullPhone – 14802000000

V. CONCLUSION

In this paper, the different kind of data integration approaches and their challenges discussed in detail. In addition, the application of XML to overcome the data integration challenges elaborated. A case study for the operational unit of an organization taken into consideration for which the different ways to apply data parsing using XML ETL elaborated. The detailed steps for all three approaches along with the sample data and validation of output data was carried out in the paper. XML is a widely used and powerful data integration tool. Due to its extra ordinary capabilities, almost all commercial ETL tools have created their dedicated services packs for XML

used for data integration. The implementation can further extend to other data integration techniques like Data virtualization & Data Federation.

REFERENCES

[1] Elisa Bertino Elena Ferrari, XML and data integration Article in IEEE Internet Computing 5(6):75 - 76, December 2001 OI: 10.1109/4236.968835

[2] Rashed Salem, Jérôme Darmont & Omar Boussaid, Active XML-based Web data integration Article (in Information Systems Frontiers 15(3):371-398 · July 2013 with 619 Reads DOI: 10.1007/s10796-012-9405-6

[3] Lucas Zamboulis and Alexandra Poulovassilis, XML Data Integration By Graph Restructuring School of Computer Science and Information Systems
Birkbeck College, University of London

[uamin Wang ; Zhiwei Ye
An ETL Services Framework Based on Metadata
2010 2nd International Workshop on Intelligent Systems and Applications
DOI: 10.1109/IWISA.2010.5473575
uamin Wang ; Zhiwei Ye

[4] An ETL Services Framework Based on Metadata
2010 2nd International Workshop on Intelligent

[5] Systems and Applications
DOI: 10.1109/IWISA.2010.5473575
Towards a Semantic Extract-Transform-Load (ETL) Framework for Big Data Integration Srividya K. Bansal 2014 IEEE International Congress on Big Data
DOI: 10.1109/BigData.Congress.2014.82

[6] Huamin Wang ; Zhiwei Ye
An ETL Services Framework Based on Metadata
2010 2nd International Workshop on Intelligent Systems and Applications
DOI: 10.1109/IWISA.2010.5473575

[6] Yue Guohua Wang Jingting
The Design and Implementation of XML Semi-structured Data Extraction and Loading into the Data Warehouse 2010 International Forum on Information Technology and Applications
10.1109/IFITA.2010.265

[7] G. Jayashree and Dr.C. Priya Design of Visibility for Order Lifecycle using Datawarehouse DOI: 10.35940/ijeat.F9171.088619

[8] G. Jayashree and Dr.C. Priya Comprehensive Guide to Implementation of Datawarehouse in Education – Proceedings of ICTIDS 2019

[9] Chong-Shan and Ran Ma-Chuan Wang An XML Schema-Based Data Integration Proceedings of the 2010 IEEE Conference 978-1-4244-5540-9/10

[10] Yue Guohua, Wang Jingting The Design and Implementation of XML Semi-structured Data Extraction and Loading into the Data Warehouse 2010 International Forum on Information Technology and Applications

[11] Nawfal El Moukhi, Ikram El Azami and Abdelaaziz Mouloudi X-ETL: a New Method for Designing Multidimensional Models 2017 IEEE Conference 978-1-5386-1115-9/17