iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** VII **Month of publication:** July 2020

DOI: http://doi.org/10.22214/ijraset.2020.7046

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VII July 2020- Available at www.ijraset.com

A Proposal of Scala Script Generation Tool for Extract Transform Load (ETL) Operations

Bhakti Deshpande[1], Samruddhi Malge[2], Samrudhi Ghorpade[3], Yashashri Bongulwar[4], Varsha Pimprale[5]
[1,2,3,4]*UG Student, Computer Engineering, Cummins College of Engineering for Women, Pune, India,* [5]*Assistant Professor, Department of Computer Engineering, Cummins College of Engineering for Women, Pune, India*

Abstract: ***ETL is a process for data integration to provide a consolidated view of data that involves three steps such as Extract, Transform and Load to combine data from multiple sources. In this process, data is fetched from numerous data sources, transformed into a particular format and finally loaded into a suitable data warehouse. Developers have to write the Scala code using Spark SQL to perform ETL operations. But as and when the requirement changes, the developers have to write code again accordingly. This paper presents an idea to build a tool to facilitate ETL process using Spark. The tool will automatically generate Scala scripts from the uploaded ETL mapping document. In order to verify the scripts, the system also provides unit test cases and SQL queries using Spark SQL. This will minimize the repetitive tasks faced by developers and provide a robust system which will ease the development of ETL operations.***

Keywords: Extract, Transform, Load, Spark SQL, Big Data, Scala scripts, ETL Mapping Document

I. INTRODUCTION

ETL (Extract, Transform, Load) is a process which provides consolidated view of data and hence makes easier for the business users to analyse and summarize the data. It involves extracting the data from the source system, transforming it in a suitable format and loading in a data warehouse. Apache Spark is a distributed technology that allows to write concise and readable code for ETL jobs. It provides a unified analytics engine for processing of large-scale data. It is a framework to analyse, process and query big data. Spark SQL is one of the components of Apache Spark that provides support for handling structured and semi structured data. Scala is a modern functional programming language that is used to handle big data using Spark and Hadoop frameworks.

Big data developers have to write the Scala code using Spark SQL to perform ETL operations. They need efforts to develop detail table and summary table. Pair programmer needs to build test cases and test scripts. But as and when the requirement changes, the developers have to write code again accordingly. The proposed system will help the developers to automate the script generation process thereby avoiding the repetitive tasks and saving their time.

This paper proposes an idea to build a tool that automatically generates Scala script from the ETL mapping document. It also generates Scala unit test cases to verify the scripts generated for different types of ETL operations. It also provides with Spark SQL queries to check for the correctness of scripts. This tool requires the user to download the standard template of excel based ETL mapping document. Based on user stories and new feature requirements, user has to prepare and upload the document. A utility will be developed to parse the uploaded ETL mapping document. The required utility will extract required fields, perform transformations and generate the Scala scripts. It will also generate Scala unit test cases and Spark SQL queries.

II. DATA FLOW DIAGRAM AND ARCHITECTURE

This section consists of the data flow diagram and architecture of the proposed script generation tool. The Data flow diagram gives the overall conceptual idea and the architecture of the system gives the detailed steps required to build the proposed tool.

A. Data Flow Diagram

Fig. 1 shows the level-0 data flow diagram showing the structure of the proposed tool as a whole and its interaction with external entities.

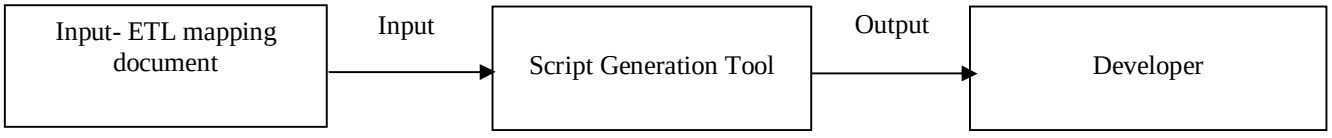


Fig. 1 Level-0 Data Flow Diagram

Fig. 2 shows the level-1 data flow diagram depicting the exploded view of the proposed system.

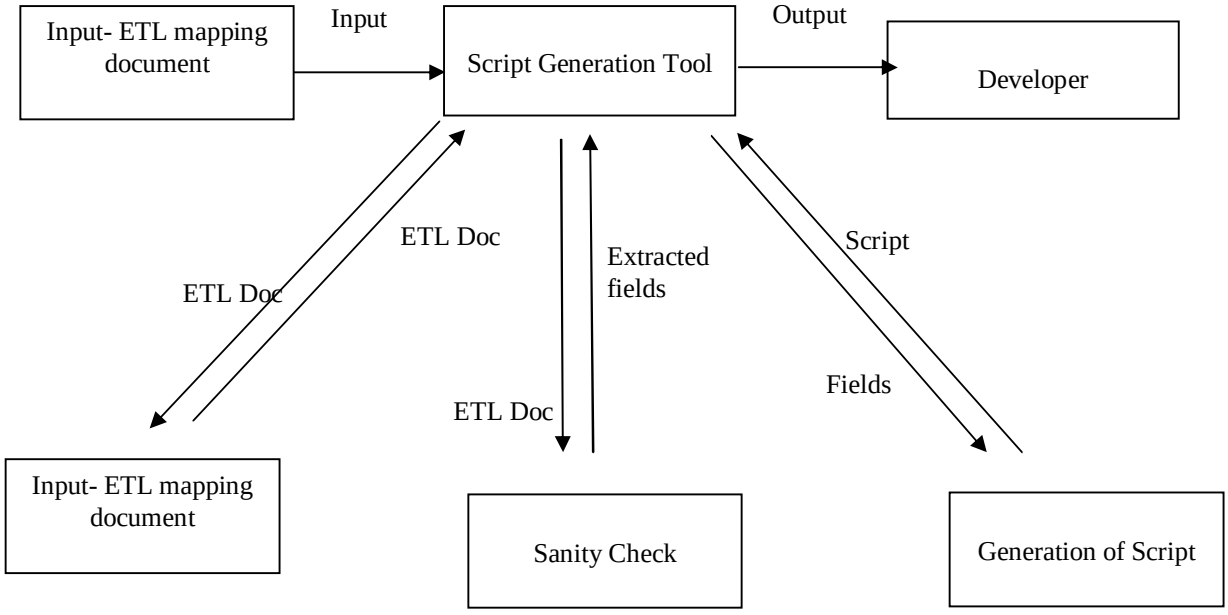


Fig. 2 Level-1 Data Flow Diagram

B. Architecture

Fig. 3 shows the architecture of the proposed script generation tool. It consists of three modules that are input module, processing module and an output module. The architecture helps to give a conceptual idea of the proposed script generation tool.

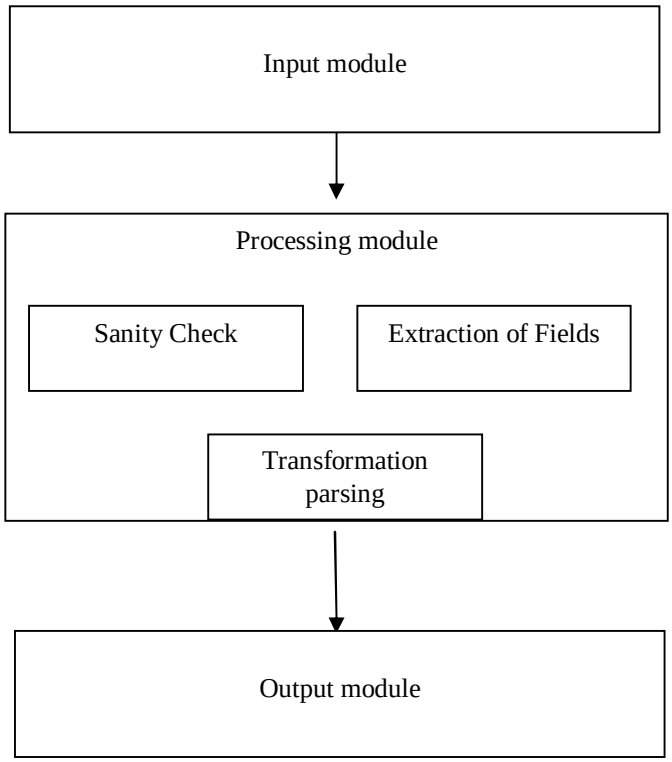


Fig. 3 Architecture of the proposed Script generation tool

The three modules proposed in the architecture are Input Module, Processing Module and Output Module. The following is the description of each module.

C. Input Module

This module takes ETL mapping document as input for the generation of Scala scripts. This document specifies required fields like source table, target table, source column, target column, data type and join condition. It also includes the necessary transformations for performing ETL operations on the data.

D. Processing Module

This module is a combination of different modules like Sanity check, Extraction of fields and Transformation parsing. Sanity check module checks whether the uploaded ETL mapping document follows the standard template. It also checks whether the ETL mapping document includes all the necessary fields like source table, target table, target column, data type and join condition. The document is rejected if any of the mentioned conditions is missing. In the Extraction of fields module, required fields like join condition, source table, target table, source column, data type and target column are extracted. In Transformation parsing module, the ETL mapping document is scanned for transformations. Each transformation is processed and parsed using string manipulation and Regular Expressions (regex). String manipulation is used to filter the transformations for further processing. The string is filtered to remove the unwanted spaces and unnecessary characters like punctuation marks. Regular expressions provide the facility to search and extract the required information from the transformations. Each transformation in the document is scanned and a regex pattern is created as per the type of transformation. The regex pattern created is matched with each type of transformation to extract the required information needed for script generation.

E. Output Module

In this module, Scala script needed for performing ETL operations is provided to the user. It also provides Scala unit test cases and Spark SQL queries.

Fig. 4 shows a sample ETL mapping document which is given as an input to the script generation tool.

Target Table	z_suresense_device_details_cid						
Source Table	tm_ssevents						
Lookup Table	z_standard						
Join Condition	z_standard(deviceid) and tm_ssevents(deviceid) inner						
Type of target table	Details						
Sr#	**Target Table**	**Target Column**	**Data Type**	**Description**	**Transformation**	**Source Table**	**Source Column**
1	z_suresense_events_details_cid	counterid	bigint	Counter ID-Auto Increment Number	Auto Increment Number	Not Applicable	Not Applicable
2	z_suresense_events_details_cid	data_as_of_date	timestamp	Spark job load time	Spark job loading time	Not Applicable	Not Applicable
3	z_suresense_events_details_cid	eventdatetime	timestamp	Time the event took place (on the device)	Computed column derived from eventdatetime Week end date for an event	tm_ssevents	eventdatetime
3	z_suresense_events_details_cid	weekdate	date	Week end date for an event (Saturday)	Direct Mapping	z_standard	companyid
4	z_suresense_events_details_cid	devicesn	varchar(256)	Device Serial Number	Direct Mapping	z_standard	devicesn
5	z_suresense_events_details_cid	deviceId	varchar(256)	Device ID	Direct Mapping	tm_ssevents	deviceid
6	z_suresense_events_details_cid	devicename	varchar(256)	Device Name	Direct Mapping	z_standard	Devicename

Fig. 4 ETL mapping document

III.PROCESS FLOW

Fig. 5 shows the process flow of the proposed script generation tool

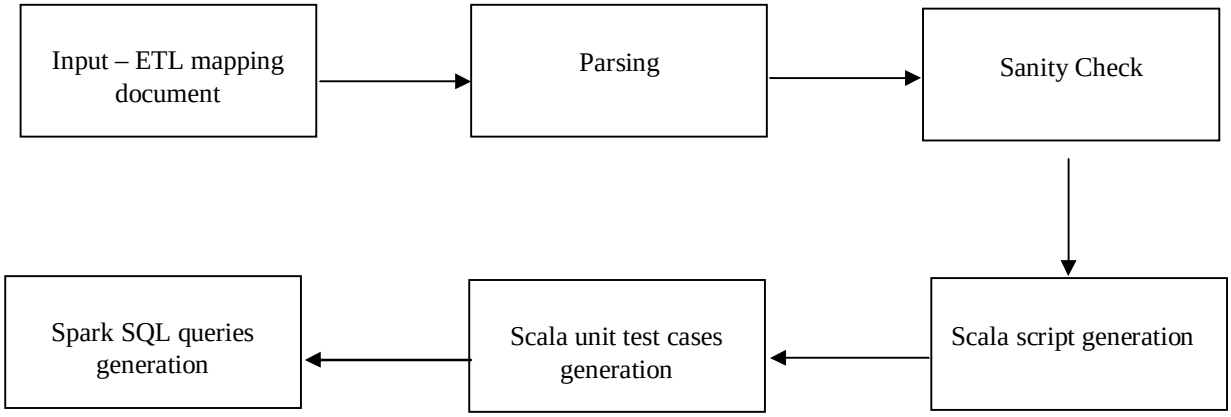


Fig. 5 Process flow of the proposed script generation tool

A. Scala unit test cases generation

This module generates unit test cases in Scala using Scala Test framework called Fun Suite and testing framework for data frames called Data Frame Suite Base. Scala unit test cases verify the correctness of scripts. These unit test cases check for equality of data frames. It also performs schema matching for comparing the schema of two data frames. It also compares the number of rows and columns in data frames. It also performs a check for data in data frames by using subtract operation.

B. Spark SQL queries generation

This module generates Spark SQL queries to check for data completeness and data correctness. Data completeness can have two values i.e. pass and fail. Data completeness is pass when the data matches exactly and fail when even a single data in two data frames is different. Data correctness can be fully match, partially match and no match. It is fully match when two data frames match each other exactly. It is partially match if some extra rows are present in either of data frames. It is no match if the data in two data frames is completely different. Data completeness and Data correctness is calculated using except operator provided by Spark SQL which is equivalent to subtract operation in set theory. Consider two data frames A and B, A-B gives the rows which are present in A but not in B. Count (A-B) gives the number rows after subtraction. Table 1 shows different conditions for data completeness and data correctness.

TABLE I
Conditions for data completeness and data correctness

Sr.no	Condition 1	Condition 2	Data Completeness	Data Correctness
1	Count (A-B) = 0	Count (B-A) = 0	Pass	Fully match
2	Count (A-B) > 0	Count (B-A) = 0	Fail	Partially match
3	Count (A-B) = 0	Count (B-A) > 0	Fail	Partially match
4	Count (A-B) > 0	Count (B-A) > 0	Fail	No match

IV.CONCLUSION

The proposed system enables generation of Scala scripts, unit test cases and Spark SQL queries for a Big Data project. The tool provides entire suite of code needed for doing Big Data Development in Spark. ETL has been a crucial method for organizations that needs to move the data from source systems to a data warehouse or another data repository from analytics purposes. This will facilitate ETL process and minimize the repetitive tasks faced by developers by providing Scala scripts and test cases.

V. FUTURE WORK

We will extend the proposed tool such that there will be no restriction on user to follow the standard template of ETL mapping document and can upload the document as per requirements. The system will also provide the script in different languages for doing Big Data operations like Python and R.

REFERENCES

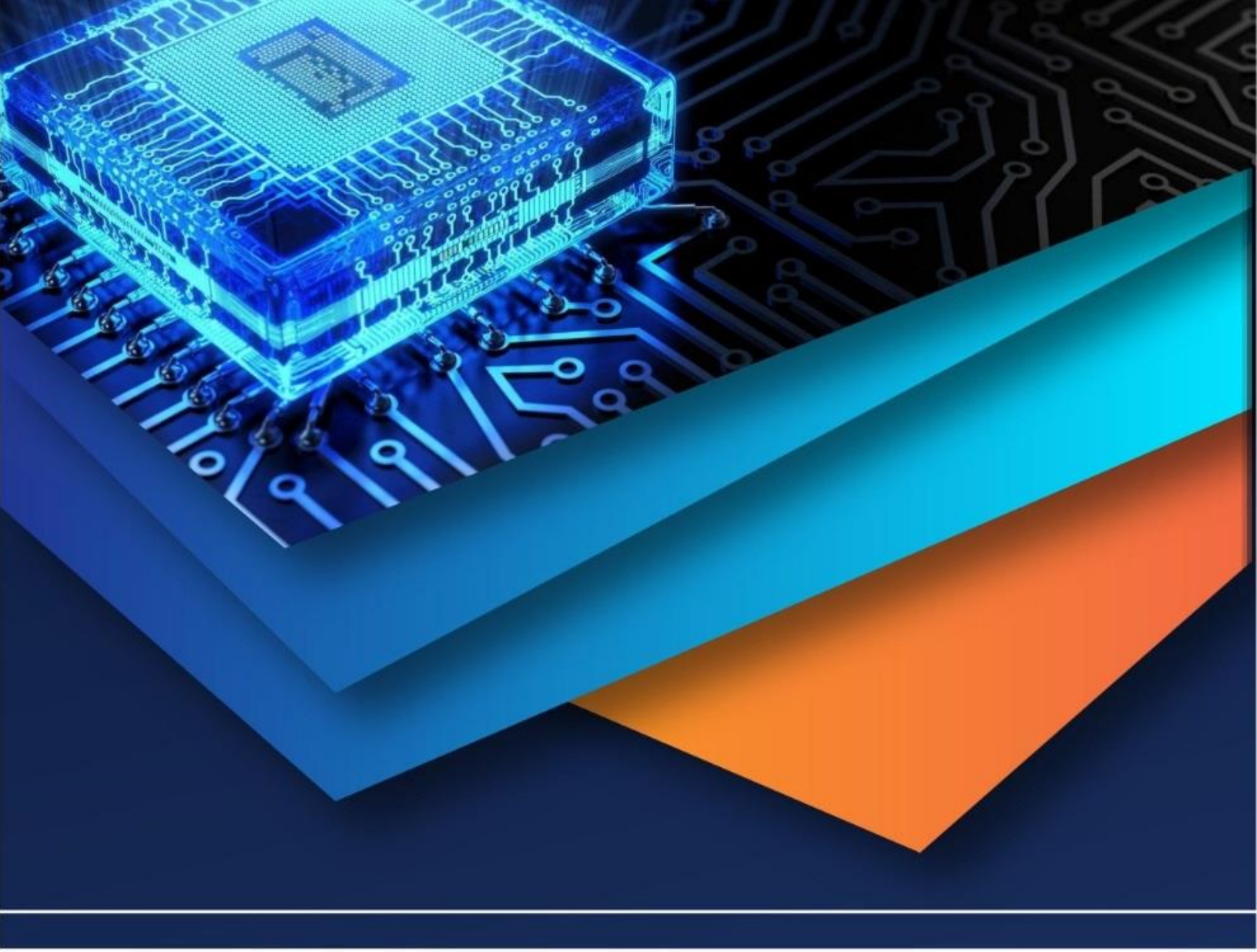
[1] Nobuo Funabiki, Ryota Kusaka, Nobuya Ishihara, "A proposal of test code generation tool for java programming learning assistant system", 31st International Conference on Advanced Information Networking and Applications, 2017.

[2] Sunil D Rathod," Automatic Code Generation with Business Logic by Capturing Attributes from User Interface via XML" IEEE International Conference on Electrical, Electronics and Optimization Techniques, Omkar Sanjay Kulkarni et al 3-5 March 2016

[3] Omkar Sanjay Kulkarni, "Automation in ETL Testing "(IJCSIT) International Journal of Computer Science and Information Technologies, 2017, Vol. 8 (6),586-589

[4] Marko Petrovic, Milica Vuckovic, Nina Turajlic, Sladan Babarogic, Nenad Anicic, Zoran Marjanovic, "Automating ETL processes using the domain-specific modeling approach", Springer-Verlag Berlin Heidelberg, 2016

[5] Abilio G. Parada, Eliane Siegert, Lisane B. de Brisolara, "Generating Java code from UML Class and Sequence Diagrams", IEEE Brazilian Symposium on Computing System Engineering,2011



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY