Database Migration from Structured Database to non-Structured Database

Mital Potey
Department of Computer Engg., AISSMS's IOIT, Pune – 411001 India

Megha Digrase
Department of Computer Engg., AISSMS's IOIT, Pune – 411001 India

Gaurav Deshmukh
Department of Computer Engg., AISSMS's IOIT, Pune – 411001 India

Minal Nerkar
Department of Computer Engg., AISSMS's IOIT, Pune – 411001 India

ABSTRACT
Database are growing very fast and becoming more complex in the volume (terabyte to petabyte), variety (structured, un-structured and hybrid), and velocity (high speed in growth). Management of database (BigData, so nomenclature) has become the global challenge. The data collection is currently managed and exploited mostly by using conventional data management tools such as classic relational database management systems (RDBMS) or conventional search engines. These systems and tools have been observed to be inefficient and incapable to handle the data in most of the cases e.g.: social networking sites. Efforts are being made to come up with some utility support in order to overcome the observed limitations. The research study was conducted to develop a Cloud-based Utility. It was aimed to convert structural database, SQL to un-structural database, NoSQL [11] systematically and sufficiently. The outcome of the study revealed that the database management becomes considerably scalable, flexible and efficient when traditional RDBMS are complimented by specifically designed rich set of alternative DBMS such as NoSQL, NewSQL based systems. The database conversion shall help in Database analysis and migration from traditional database to today's BigData.

General Terms
SQl, NoSQL, Database, Python, Django, BigData.

1. INTRODUCTION
Many organizations collect vast amounts of customer, scientific, sales, and other data for future analysis. Traditionally, most of these organizations have stored structured data in Relational Databases for subsequent access and analysis. However, a growing number of developers and users have begun turning to various types of Non-Relational now frequently called NoSQL databases.

Relational databases such as MySQL has predefined schemas fixed table names and types of columns and Relational databases don't work easily in a distributed manner because joining their tables across a distributed system is difficult.

NoSQL databases have dynamic schemas. NoSQL generally process data faster than relational databases. In most typical situations, SQL databases are vertically scalable. You can manage increasing load by increasing the CPU, RAM, SSD, etc. on a single server. On the other hand, NoSQL databases are horizontally scalable. You can just add few more servers easily in your NoSQL database infrastructure to handle the large traffic.

To get a gist of working of the database migration, the migration program will be written in Python [1] [2] language which will be deployed on Amazon Web Service EC2. Web interface will be developed using Django Framework for uploading and downloading the converted database. Parallel computing is used to enhance the speed of computing.

his paper will give a brief introduction to all the tools and programming language used for database migration. It also talks about an optimization method to increase the speed of migrating huge data from one format to another.

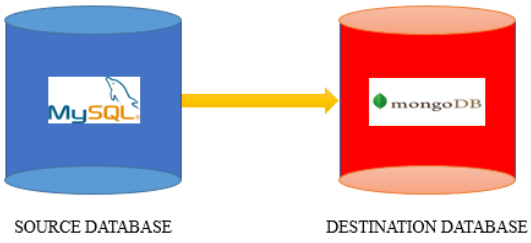

Fig 1: Problem Statement: Database Conversion from SQL to NoSQL

Problem Definition
There's nothing wrong with the traditional RDBMS. It simply isn't enough for the application dealing with huge databases. Also, NoSQL databases need cheap hardware. Hence, some of the MySQL databases need to be converted to NoSQL databases. Many institutions and big companies need to convert their big databases (MySQL).Our utility will help them to convert from MySQL to NoSQL [9] [10] [12].

2. RELATED WORK
Last decade records a huge amount of data transition taking place on the internet. Today's world is connected by social networking sites. Such sites have huge sizes of databases. Our review in this area shows that not many approaches that provide tools or utilities for database conversion.

Ton Blankers in [3] proposed discusses trends, challenges and solutions of data conversion in application modernization projects. An utility for converting one SQL database into another SQL database was proposed in this paper.

The white paper published by Oracle [4] mentions techniques and utilities for migrating non oracle databases to oracle databases.

Abdelsalam Maatuk in [5] describes an investigation into approaches and techniques used for database conversion. Constructing object views on top of a Relational DataBase (RDB), simple database integration and database migration are among these approaches.
Andreas Meier in [6] reviews and describes data and code conversion to assist present applications in running entirely on new generation database technology.
A B M Moniruzzaman in [7] aims at providing - classification, characteristics and evaluation of NoSQL databases in Big Data Analytics.
Though there isn't any available utility converting NoSQL databases into SQL databases, Sanobar Khan in [8] attempts to use NoSQL database to replace the relational database. It mainly focuses on one of the boosting technology of NoSQL database i.e. MongoDB, and makes a comparison with MySQL and thus justifies why MongoDB is preferred over MySQL.

Currently there is no utility present for converting SQL databases into NoSQL databases. As NoSQL is a new trend, there are no software or utilities that are designed for the desired conversion. Most of the utilities developed were developed earlier for converting one form of SQL database to another SQL database.

The significance of our methodology is that we provide an online utility that helps to convert MySQL database to types of NoSQL databases such as MongoDB. Not only is our methodology according to today's professional requirements but is also launched using cloud services which offer other services like scalability, elasticity and high availability of data.

3. PROPOSED SYSTEM

In the previous section, we have reviewed the software interface required for our utility to function. We have proposed a utility that will be working on Amazon cloud utilizing the Amazon Cloud Services. It provides for an end user to register; purchasing the subscription of the utility in which the user can later upload his SQL database and get a downloadable copy of the converted NoSQL version of his database. Proposed System is divided into 3 main steps/modules as follows.
Module 1 is User Registration,
Module 2 is Database Uploading/Downloading,
Module 3 is Database Conversion wherein lies the main logic of database conversion.

User Registration

In this module the user is supposed to register for accessing the database conversion utility on Amazon Web service and pay the relevant purchase amount as decided by the organization. We will be providing a simple GUI for User Registration which will be designed in Django framework. There will also be an administrator login for managing all the registrations done on the server.

Database Uploading/Downloading

In this module the user will be provided with a simple and easy to understand GUI for uploading his SQL database and downloading his converted NoSQL database. User has to enter in the type of database he will be uploading and the database he wants his SQL database to be converted into.
The uploading of the database facilitates the server freedom and enhances the efficiency. This is due to the reason that once the database has been uploaded, the server remains free for fetching of another database. Also, once the database has been uploaded, the speed of the conversion increase dramatically as the database is accessed by the server from the server and not over the internet.

Database Conversion

This module comprises of the main logic of the Database conversion.
It is mainly divided in following step:
i)Database Validation : The first step of the conversion is to validate the entered SQL database. This will be done by checking the schema of the database and the relevant criteria of the selected SQL database.
ii)Database Schema Extraction : The second step is to extract the schema of the database entered. This includes extracting the names of the tables, the respective columns, the datatypes of the columns, etc. This schema extraction will be provided as an input for the formation of the NoSQL database.
iii)Database Schema Creation : The third step is to create the schema for the new NoSQL database. The schema extracted in the previous step is used for this purpose. The new NoSQL database is formed according to the schema of the NoSQL database selected by the user. This varies for every database. For e.g. The conversion to MongoDB consists of considering every row of the SQL database as a document and pushing it in the MongoDB schema which is made by the extraction of the column names from the SQL database.
iv)Database Deployment : The last step of the conversion process is to deploy the converted database. The user will be able to download the converted database from the server once the conversion has been completed. This allows for user to be less dependent of continuous internet access. The user only needs to have a continuous internet access for uploading the database. He can download the converted database anytime needed.

4. COMPARITIVE ANALYSIS AND RESULT

Designing the above mentioned system, we did some performance evaluation tests. Though the database sizes used for the analysis is comparatively smaller compared, we do get a clear difference in the various factors of comparison. We converted the some databases by manually extracting data from SQL databases and then creating NoSQL databases. We also converted the same databases using some of the methods using heuristic methods and coding. And later we tested our utility's performance under similar circumstances. The following table describes the observed and analyzed results.

Table 1. Performance evaluation between manual approach, heuristic solutions and suggested approach

Sr. No.	Factors	Manual Approach	Heuristic Solutions	Suggested Approach
1	Efficiency	Less	Good	Very Good
2	Human Effort	More	Moderate	Less
3	Resource Utilization	Less	Moderate	High
4	Error Scope	More	Less	Less

5. CONCLUSION

If computational and storage requirements of applications such as for Big Data Analytics, Business Intelligence and social networking over peta-byte datasets have pushed sql-like centralized databases to their limits. This led to the development of horizontally scalable, distributed non-relational No-SQL databases. NoSQL databases are now getting attention of application developers due to the following reasons:
NoSQL databases provides schema-less dynamic flexible data model, that is most suitable for the big users and big data.
NoSQL databases have an ability to scale dramatically to support global users and big data.
NoSQL databases provide an improved performance to satisfy big user's expectation without compromising scalability.
Thus in our project NoSQL database approaches to supporting applications that process huge volumes of data as well as to provide a global overview of this non-relational NoSQL databases.

6. REFERENCES

[1] Multi-paradigm programming language. developer.mozilla.org, 21 October 2013.

[2] Python 3.4.1. Python Software Foundation, 21 May 2014.

[3] Tom Blankers, Uniface Client Manager, Data Conversion for Modern Applications, 2014

[4] Migrating Applications and Databases with Oracle Database 12*c*, An Oracle White Paper, June 2013

[5] Relational Database Migration: A Perspective, School of Computing, Engineering & Information Sciences, Northumbria University, Newcastle upon Tyne, UK

[6] Providing Database Migration Tools A Practitioner's View

[7] A B M Moniruzzaman and Syed Akhter Hossain, NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison, International Journal of Database Theory and Application Vol. 6, No. 4. 2013

[8] Sanobar Khan and Prof.Vanita Mane, SQL Support over MongoDB using Metadata International Journal of Scientific and Research Publications, Volume 3, Issue 10, October 2013

[9] R. Lawrence. Integration and virtualization of relational sql and nosql systems including mysql and mongodb. In Computational Science and Computational Intelligence (CSCI), 2014 International Conference on, volume 1, pages 285{290, March 2014.

[10] N. Leavitt. Will nosql databases live up to their promise? Computer, 43(2):12{14, Feb 2010.

[11] Yishan Li and S. Manoharan. A performance comparison of sql and nosql databases. In Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on, pages 15-19, Aug 2013