Bridge From Mysql to Nosql

A Project Report Submitted
in Partial Fulfillment of Requirements
for the Degree of

Bachelor of Technology

by

Akshay Prasad(2012CSB1005)
P. Jaya Bharath (2012CSB1025)

Department of Computer Science & Engineering

Indian Institute of Technology Ropar

Rupnagar 140001, India

April 2012

Abstract

The demand for Datastorage and accessing speed of the stored data has been increasing day by day with the growth in the number of users. With the increase in demand there is development of New methods, ways of storing data. Some of the most famous ways are MySql and NoSql. Since the starting, structured data has always been used to stored data. There are a number of relational databases that are used in these applications to store required data for example MySQL, PostgreSQL, Helix database, SQLBase, Microsoft SQL Server etc. Although relational databases are widely used in applications and working on relational databases are very simple, developers are shifting from traditional relational databases to NoSQL databases. We thought it would be good if we were able to come up with a method which can help us in converting the data stored in one format to the other, so that we can use the required way of data as per our requirement. In the Process, We have built a Tool to act as bridge from Mysql to Nosql for 1. MongoDb 2. Google DataStore.

Acknowledgements

We would like to express our special thanks to our Course advisor and our guide Dr. Balwinder Sodhi. His guidance and constant support encouraged us to accomplish the task that we have taken a year ago. we learned to build tools that could be used in real world applications. We were able to explore concepts which were used at industry level because of this project. we learned how to compared available tools and to choose which one should we select basing on our requirements. we realized that how useful it is to build tools which can migrate the existing format to a new and more useful format without any loss of data

Certificate

It is certified that the B. Tech. project “Schema translation and bridge from SQL to NoSQL ” has been done by Akshay Prasad(2012CSB1005), P. Jaya Bharath (2012CSB1025) under my supervision. This report has been submitted towards partial fulfillment of B. Tech. project requirements.

Dr. Balwinder Sodhi
Project Supervisor
Department of Computer Science & Engineering
Indian Institute of Technology Ropar
Rupnagar-140001

Honor Code

We certify that we have properly cited any material taken from other sources and have obtained permission for any copyrighted material included in this report. We take full responsibility for any code submitted as part of this project and the contents of this report.

Akshay Prasad (2012CSB1005)

P. Jaya Bharath(2012CSB1025)

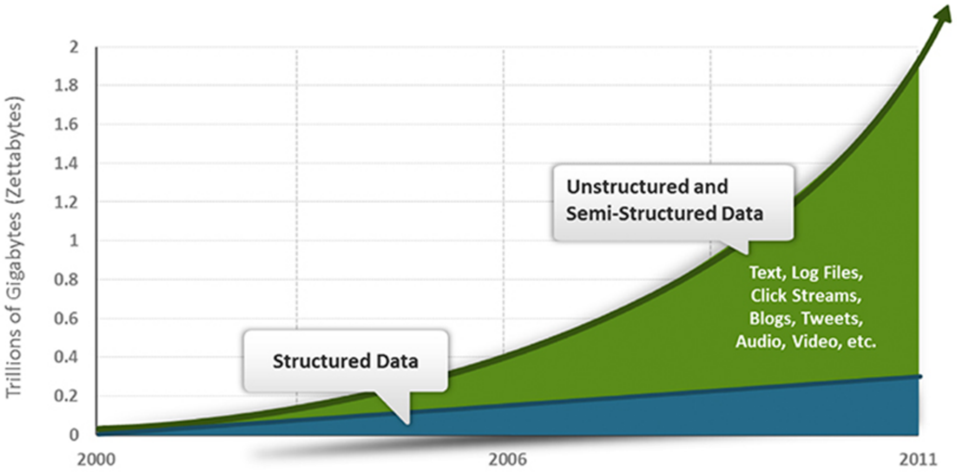
Contents

List of Tables

Chapter 1

Introduction

In computer applications, mostly web based, data storage is a crucial part. Since the starting, structured data has always been used to stored data. There are a number of relational databases that are used in these applications to store required data for example MySQL, PostgreSQL, Helix database, SQLBase, Microsoft SQL Server etc. Although relational databases are widely used in applications and working on relational databases are very simple, developers are shifting from traditional relational databases to NoSQL databases. One of the reason for this is limitation of scalability in SQL databases. SQL Databases are vertically scalable this means that they can only be scaled by enhancing the horse power of the implementation hardware, thereby making it a costly deal for processing large batches of data. Also, with increasing size of the database or increasing number of users, Relational Database Management Systems using SQL suffer from serious performance bottlenecks -making real time unstructured data processing a hard row to hoe. NoSQL is a database technology driven by Cloud Computing, the Web, Big Data and the Big Users. NoSQL Database, also known as Not Only SQL is an alternative to SQL database which does not require any kind of fixed table schemas unlike the SQL. As the number of users increase for some application, developers tend to use NoSQL databases. There is a need for proper tool to transfer data from SQL to NoSQL in order to continue the working of the application without any data loss. So, we decided to make a tool for migration of SQL to NoSQL databases and corresponding queries.



Source: IDC 2011 Digital Universe Study (http://www.emc.com/collateral/demos/microsites/emc-digital-universe-2011/index.htm)

Figure 1.1:

Graph 1.1 is a survey Graph we Got online, its shows the shift from MySql to Nosql Over years.
It is basically showing the Increasing Demand for Unstructured, Semi Structured Data.

Chapter 2

Introduction

2.1 Initial Procedure and Basics

First task is to extract data types from provided SQL file for each Table present in the SQL file from CREATE TABLE queries and to create **DTO** and **DAO java classes** for corresponding to each Table.

1. DTA class contains Java data types corresponding to SQL data types for each of each column of a Table.
2. DAO class includes methods to perform SQL operations on NoSQL database by converting SQL queries to NoSQL filter and operators and Java data types into NoSQL data types.

As we are going to Convert From MySql to NoSql let us look have a overview of both the platforms that we are operating on.

SQL Databases

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language, data manipulation language, and a data control language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. SQL is used for Operations on relational data bases. A relational database system contains one or more objects called tables. The data or information for the database are stored in these tables. Tables are uniquely identified by their names and are comprised of columns and rows. Columns contain the column name, data type, and any other attributes for the column. Rows contain the records or data for the columns.

1. Store data in a well structured relational model with rows and columns.
2. Each record have a fixed schema(the columns must be decided and locked before data entry and each row must contain data for each column)
3. Scaling is vertical. More data means bigger server, which can be expensive.

NoSql Databases NoSQL systems are also called "Not only SQL" to emphasize that they may support SQL-like query languages. NoSql Provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Motivations for this approach include: simplicity of design, finer control over availability. Data structures used by NoSQL databases (e.g. key-value, wide column, graph, or document) differ slightly from those used by default in relational databases, making some operations faster in NoSQL. Most NoSQL databases offer a concept of "eventual consistency" in which database changes are propagated to all nodes "eventually" (typically within milliseconds) so queries for data might not return updated data immediately or might result in reading data that is not accurate.

1.Each NoSQL database follow different data models e.g., document, graph, key-value etc.
2.Schema is dynamic.
3.Scaling is horizontal, meaning across servers. Multiple servers can be cheap commodity hardware or cloud instances, making it a lot more cost-effective than vertical scaling.
Every company/organisation can choose their own NoSql Structure.
Given the Time constraint we have selected to work on two major platforms
1. MongoDb 2. Google DataStore

In the Table we have compared The Structures of Chosen Platforms.

Table 2.1:

SQL (MySQL)	NoSQL (Google Datastore)	NoSql(MongoDB)
Table	Collection	collection
Column	Field	Field
Row	Document	Document
Primary Key	Primary Key	Primary Key

Table 2.2:

Sql Comparision Operator	Datastore Comparison Operator	Meaning
=	$eq	Equal to
<	$lt	Less than
<=	$lte	Less than or Equal to
>	$gt	Greater than
>=	$gte	Greater than or equal to
!=	$ne	Not Equal to
IN	$in	Member of

2.2 Working and overview of the Programme

The code will take input SQL file and create DAO and DTO class files for each CREATE TABLE statement present in SQL file. User will have to include both DAO and DTO class files in his/her project. After including generated DAO and DTO files in project folder, user should create objects of DAO file and pass the required query to corresponding methods. DAO file generated by the code contains public variable specified for each table e.g. Table name, columns of table, Data types of columns, SQL and Datastore comparators, primary key, etc. Along with variables there are methods user can call for performing queries. insert : This method insert data, provided in SQL query, to Datastore. Select : This method extract data, provided in SQL query, from Datastore and provide them to user in the form of corresponding DTO objects. Delete : This method delete data, using provided SQL query, from Datastore. Update : This methods update data, using provided SQL query, in Datastore

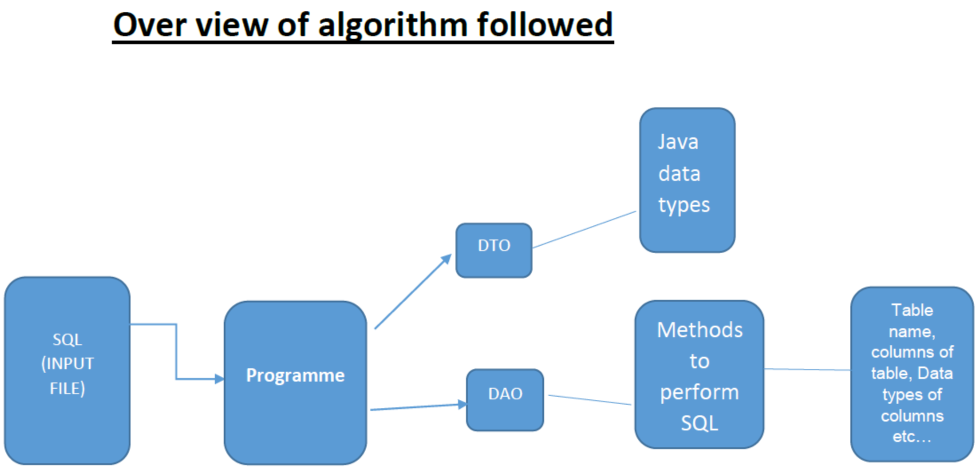

Over view of algorithm followed
SQL (INPUT FILE)
Programme
DTO
Java data types
DAO
Methods to perform SQL
Table name, columns of table, Data types of columns etc…

Figure 2.1:

Chapter 3

MongoDb

3.1 Recommended Installations and Basics

ere we are discussing about MongoDB
MongoDB is a cross platform oriented document oriented database. As of July 2015, MongoDB is the fourth most popular type of database management system, and the most popular for document stores.

Installations

1. **Eclipse**
2. **MongoDb Plugin**
3. **Java Plugin.**

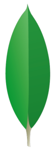


Figure 3.1:

3.2 Method and Approach

First task is to extract data types from provided SQL file for each Table present in the SQL file from CREATE TABLE queries and to create DTO and DAO java classes for corresponding to each Table.
DTA class contains Java data types corresponding to SQL data types for each of each column of a Table.
DAO class includes methods to perform SQL operations on NoSQL database by converting SQL queries to NoSQL filter and operators and Java data types into NoSQL data types.

Let us look at the Five Basic Queries
1. **SELECT**
Select query is used to extract data from Database:-
By primary key
By comparators on different columns
2. **DELETE**
Delete query is used to delete data from Database:-
By primary key
Providing comparators for one or more columns
3. **UPDATE**
Update query is used to update previous data in Database.:-
By primary key
By comparators on other values of columns
4. **INSERT**
Insert query is used to insert data into Database
5. UNION(PROVIDE two queries separately as string)

3.3 Clauses and Syntax

Clauses are very important in getting our results efficiently. Using the correct Clause will help to get the results Quickly and more efficiently

Clauses that are supported

1. **Where**
2. **Order by**
3. **Offset**
4. **Group by**

Syntax to be followed (We have decided to go with a specific syntax)
Column name then Comparator then its Value either in form of value or subquery

Conditions Acceptable
AND - [Equality","Less Than","Less Than Equals","Greater Than","Greater Than Equals","Not Equals] - OR,IN,BETWEEN
The above Example shows that after we get an AND the next Operator can be anyone belonging to the set given then followed by OR/IN/BETWEEN.
ORDER BY : Scrictly column name to oder by.
OFFSET : Stritcly number to sort.
GROUP BY : striclty column name ::used when we use avg,sum,count,etc..

Join Queries are Not Accepted
Example queries

3.4 Examples

```
1. SELECT firstname,lastname, MIN(actorid), COUNT(firstname) from actor
where actorid<100 GROUP BY lastname
2. SELECT name FROM category WHERE categoryid<10
3. SELECT address FROM address
```

Chapter 4

Google DataStore

4.1 Recommended installations and Basics

1. Eclipse.
2. Java Plugin.
3. Google Datastore Plugin.

In this part we are discussing about Google Data Store

Google Datastore is a NoSQL document database built for automatic scaling, high performance, and ease of application development.

Datastore features include:
Atomic transactions, High availability of reads and writes, Encryption at rest.

Figure 4.1:

4.2 Method and Approach

First task is to extract data types from provided SQL file for each Table present in the SQL file from CREATE TABLE queries and to create **DTO and DAO java classes** for corresponding to each Table.
1. DTA class contains Java data types corresponding to SQL data types for each of each column of a Table.
2. DAO class includes methods to perform SQL operations on NoSQL database by converting SQL queries to NoSQL filter and operators and Java data types into NoSQL data types.

The four basic queries that were supported are

1.select Select query is used to extract data from Database.:-
By primary key.
By comparators on different columns.

2.delete Delete query is used to delete data from Database
By primary key
Providing comparators for one or more columns.

3. update Update query is used to update previous data in Database.
By primary key.
By comparators on other values of columns.

4. Insert Insert Query: Insert query is used to insert data into Database.

There are few more essential methods included which are not to be altered. If user modifies these methods, they can lead to errors. User must have Google Datastore sdk to use these generated java scripts.
These scripts interact with Datastore locally. User has to upload the application to Google server later.

4.3 Clauses and Syntax

Some of the clauses that were supported are
1. Where 2. Order by 3. Offset 4. Group BY (This is not supported in google datastore) 5. Sort By

A Note on Syntax Being Followed
1. Column name then comparator then its value in form of value.
2. ONLY one inequality condition is accepeted by Google datastore rest must be equalites
3. All the basic operators are supposed to be in capitals like "SELECT", "WHERE" Etc...

Chapter 5

Conclusions

The tool developed by us is providing desired output to the following open source databases
1. Sakila database.
2. World database.
3. Lahman database
4.India index database.

1. There are no additional plugin used.
2. The code is well documented and easily modifiable if required.
3. If some new data types and/methods come in future they can be added to code without much modification as it uses properties file to store variables which tends to change or modify.

References

http://www.tutorialspoint.com/mongodb/
Google Datastore page:- https://cloud.google.com/datastore/
Datastore tutorial page:- http://googlecloudplatform.blogspot.in/2013/05/get-started-with-google-cloud-datastore-nosql-database.html GAE docs:-
https://cloud.google.com/appengine/docs Paper regarding SQL to NoSQL transformation:-

http://link.springer.com/article/10.1007/s10515-013-0135-x MySQL website
https://www.mysql.com/ Java docs
http://docs.oracle.com/javase/7/docs/api/ Datastore links
https://cloud.google.com/datastore/docs/concepts/overview
https://cloud.google.com/appengine/docs/java/datastore/entities
https://cloud.google.com/appengine/docs/developersconsole/datastoreadmin?csw=1backup$_and_r$
$https : //cloud.google.com/datastore/docs/concepts/indexes$
$https : //cloud.google.com/datastore/docs/concepts/queries$
$http : //www.tutorialspoint.com/mongodb/$
$https : //www.mongodb.com/compare/mongodb - mysql$
$http : //blog.mlab.com/2012/08/why - is - mongodb - wildly - popular/$
$https : //www.google.co.in/search?q = mongodb+iconbiw = 1366bih = 667tbm =$
$ischimgil = vA3PKZ7N8AYc7M$
$https : //www.google.co.in/search?q = google + app + engine + iconbiw =$
$1366bih = 667tbm = ischimgil = CwQL1bn9XGrkRM$