General Research on Database Migration from RDBMS to HBase

Chen Liu[1, a],Zhicheng Fu[2, b], Zhengqiu Yang [2,c] and Jiapeng Xiu[2,d]
[1] School of Computer Science Beijing University of Posts and Telecommunications Beijing, China
[2] School of Software Engineering Beijing University of Posts and Telecommunications Beijing, China
[a] liuchen@bupt.edu.cn, [b] fuzhicheng@bupt.edu.cn, [c]zqyang@bupt.edu.cn, [d]xiujiapeng@bupt.edu.cn

Abstract

In an information-based world, there needs the support of big data. But the emergence of large data makes the various shortcomings of RDBMS exposed. The application of distributed database HBase can solve this problem. Based on HBase, a very popular distributed database, this paper proposes a method to migrate database which is on the huge amounts of data from RDBMS to HBase and demonstrates the realization of this semi-automatic migration design method in the process of data migration .

Keywords: database migration; RDBMS; HBase

1 Introduction

With the information explosion on the Internet, Web2.0[1] era has come, and the first challenge that comes along is the massive data processing[2] problem. For developers, complex Web 2.0 applications mean the higher demands for rational application architecture, robustness, and operation performance. In the case of big data and high concurrent access, how to make the database provide stable and reliable performance has become an important issue to be solved in current Web 2.0 applications. Since the 1970s, relational database management systems (RDBMS) have dominated the data landscape. But as businesses collect, store and process more and more data, relational databases are harder and harder to scale. More and more developers tried to abandon traditional relational database to explore other solutions[3].

In the past few years we have seen a veritable explosion in various ways to store and retrieve data. The so-called NoSQL(not only structured query language) databases have been leading the change and creating all these new persistence choices. NoSQL is a general term for non-relational database[4]. Compared to traditional relational database, NoSQL database has four advantages, that is easy to extend, high performance, flexible data model and high availability. NoSQL database removes the relational features, so the database expansion becomes easy. NoSQL database has good performance in reading out and writing in big data because of its simple database structure. On the premise of the performance not affected, NOSQL database can easily achieve a high availability architecture. Therefore, NoSQL database is more suitable for processing big data.

HBase is one of the most popular NoSQL databases, and it is modeled deriveing from Google's BigTable[5]. HBase is a component of the Hadoop[6] project, as well as a distributed, Column-Oriented open source database. It has the approximately optimal writing performance and excellent reading performance, can make the I/O utilization reach saturated. HBase provides a store view different from common two-dimensional table, the columns of Hbase table can be dynamicly adjusted on demand ,and the records in table are organized in column other than in row. It is because of HBase database has a huge advantage, more and more web applications are beginning to try to rebuild the relational database replaced by HBase when traditional RDBMS appears performance bottlenecks on high concurrent access and large amount of data access. In the application of HBase, there inevitably involves the data migration problem, that is, migrates the legacy data from the original RDBMS to HBase. How to make the original data which stored in RDBMS migrate to HBase becomes a very hot issue nowadays.

2 Key Points of Database Migration

EPPCS (Electronic Product price comparison system) is an e-commerce system which provides electronic product discounts and promotions information through using crawler tools to crawl electronic product information in major domestic e-commerce websites. The data layer module of this system had originally used MySQL to store products data. But with nearly 200 thousand database records produced every day, MySQL has resulted in low efficiency of writing, storage and reading performance problems under the circumstances of big data written in and read out. Thus, we need to migrate the database from MySQL to HBase.

3 Data Model of HBase

HBase’s data model is very different from what we have likely worked with relational databases . It is a sparse, distributed, persistent multidimensional sorted map, which is indexed by a row key, column key, and a timestamp.

A table in HBase would look like Table.1, and it’s easier to understand the data model as a multidimensional map shown in Fig.1

Table.1 A table structure in HBase

Row Key	Column Family1		Column Family2		
	Column1	Column2			
001	Data1	Data2			

```
Table{
"Row-Key1" : {   //Row key
  "ColumnFamily1" : {  //Column Family
    "Column-Key1"：{ value1 }//Column Quallifier
    "Column-Key2"：{ value2 }
     …
     }
 }
}
```

Fig.1. One row in an HBase table represented as a multidimensional map

4 Conversion of Table Schema

Unlike relational database, there is no association relationship between HBase tables, the join operation isn't supported in HBase. One way to meet the join query requirements of HBase table design is to make associated data stored in one row. So if we want to meet the join query requirements after migration, we need to convert the table schemas according to the original table mapping relationships. Usually, there are four conversion ways.

Basic Convert：copy the table schemas directly from relational database into HBase database, that is relational database table name act as HBase table name and column family name, columns act as HBase columns. This is the most simple and most fundamental transformation.

Nest Convert: among the database table A, table B and table C, there has one-to-one, one-to-many or many-to-many mapping relationships. For example: table A references to table B and table C. Thus, remain table A unchanged and split table B and table C, then append columns of B and C to table A as its new columns separately and delete table B and C finally.

Split Convert: among the database table A, table B and table C, there has one-to-one, one-to-many or many-to-many mapping relationships. For example: table A references to table B and table C references to table B, either. Thus, remain table A and table C unchanged and split table B, then append columns of B to table A and table C separately as their new columns and delete table B finally. This is the reverse of nest convert.

Inline Convert: among the database table A, table B and table C, there has one-to-one, one-to-many or many-to-many mapping relationships. For example: table A references to table B and table B references to table C. In this case, use nest convert method as well to append columns of C to table A as its new columns while considering transitive relation from A to C.

5 Process of Database Migration

The process of database migration can be divided into following several steps:

a. Obtain the relational database and HBase database connection information.

b. According to the connection relations, define the mappings between HBase database and relational database, then write the mappings into a mapping XML file and HBase table schema definition into a HBase schema XML file.

c. Establish HBase tables based on the HBase schema definition file.

d. Migrate data from relational database table to corresponding HBase table based on the mappings.

6 Design and Realizations of Database Migration

Design of System Module

The database migration system mainly consists of the following modules:

1) The RDBMS manager

This module is used to establish the connection with source database and save the table schemas of relational database into a XML file.

2) The table schema convertor

This module converts database table schemas from RDBMS to HBase according to the above table schema conversion ways based on mapping relationship between tables.

3) The table schema adapter

This module is used to save and read the table schema definition of the specified file processed by the table schema convertor.

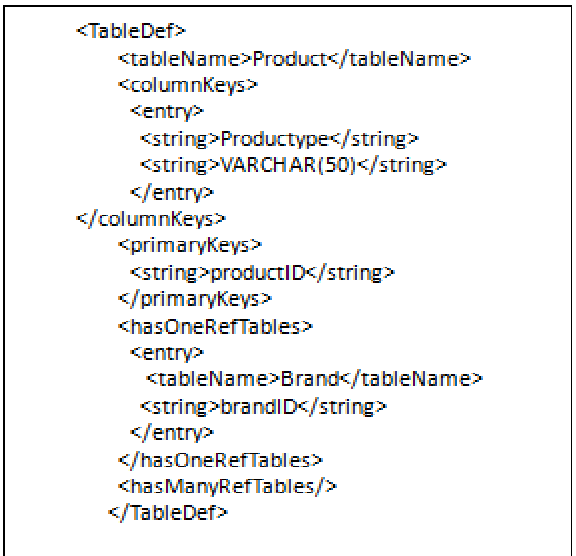
4) HBase table manager

This module is used to establish HBase tables according to the table schema definition of the HBase table schema XML file, then migrate data of RDBMS database table to HBase database according to convert-mapping XML file.

7 Realizations of key modules

1） Realizations of the RDBMS manager

The RDBMS manager reads MySQL configuration information from config.properties file,then use JDBC connect API to connect MySQL database and access the table schemas via JDBC DatabaseMetaData. Finally, write the RDBMS table schemas into XML file in specified format. The format of RDBMS table schemas XML file is shown in Fig.2.



```
<TableDef>
    <tableName>Product</tableName>
    <columnKeys>
      <entry>
        <string>Productype</string>
        <string>VARCHAR(50)</string>
      </entry>
</columnKeys>
    <primaryKeys>
      <string>productID</string>
    </primaryKeys>
    <hasOneRefTables>
      <entry>
        <tableName>Brand</tableName>
        <string>brandID</string>
      </entry>
    </hasOneRefTables>
    <hasManyRefTables/>
   </TableDef>
```

Fig.2 RDBMS table schemas XML file

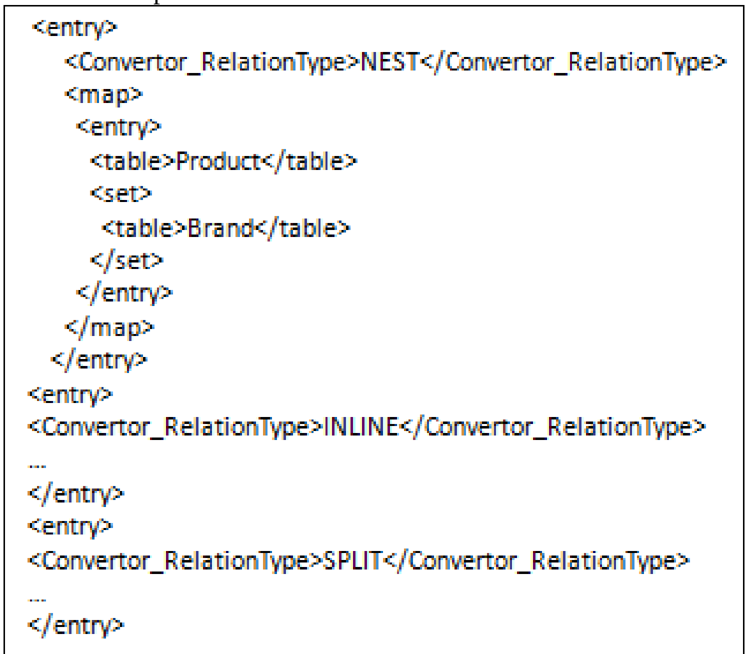
2）Realizations of the table schema convertor

The table schema convertor is the key part of the system,we could achieve it via following steps.

Step1: read and parse the MySQL table schemas XML file,get all the mapping relationships and put them in mapping set.

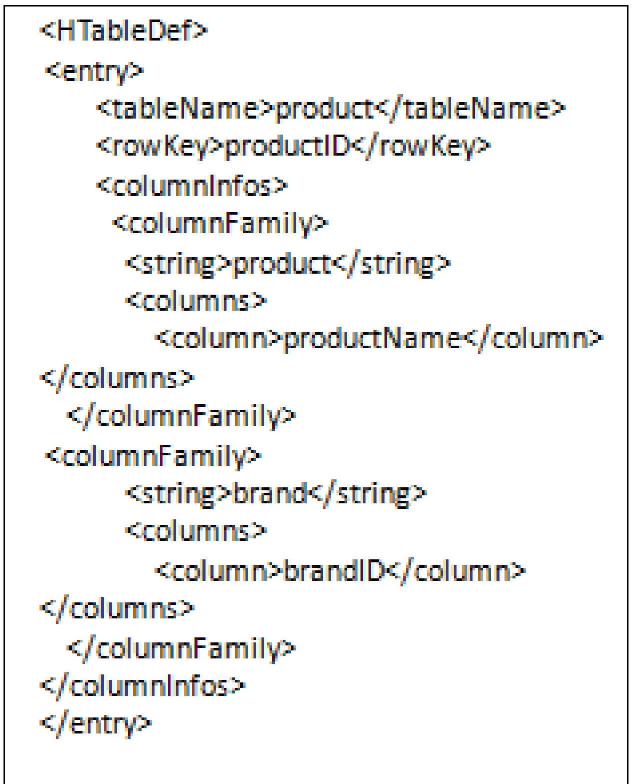
Step2:fetch one mapping relationship from mapping set in turn.According to each of the mapping relationships,find out all the database tables which satisfy nest,split or inline conversion.Finally, write the convert relation mappings into convert-mapping XML file in specified format.The format of convert-mapping XML file is shown in Fig.3.

Step3:According to the convert relation mappings,generate the Hbase table schemas. Write the Hbase table schema into Hbase table schemas XML file in specified format.The format of Hbase table



```
 <entry>
     <Convertor_RelationType>NEST</Convertor_RelationType>
     <map>
       <entry>
         <table>Product</table>
         <set>
           <table>Brand</table>
         </set>
       </entry>
     </map>
   </entry>
<entry>
<Convertor_RelationType>INLINE</Convertor_RelationType>
...
</entry>
<entry>
<Convertor_RelationType>SPLIT</Convertor_RelationType>
...
</entry>
```

Fig.3 Convert-mapping XML file



```
<HTableDef>
 <entry>
      <tableName>product</tableName>
      <rowKey>productID</rowKey>
      <columnInfos>
        <columnFamily>
         <string>product</string>
         <columns>
            <column>productName</column>
</columns>
   </columnFamily>
 <columnFamily>
         <string>brand</string>
         <columns>
            <column>brandID</column>
</columns>
   </columnFamily>
</columnInfos>
</entry>
```

Fig.4 Hbase table schemas XML file

schemas XML file is shown in Fig.4.

8 Conclusion

At present, The database migration from RDBMS to HBase database is quite inconvenient, which makes the entire migration work complicated and time consuming. However it is the problem that a lot of web 2.0 applications have to face during the expansion of database service. In order to solve this problem, this article proposes a semi-automatic database migration design method to help developers reduce the complexity of the migration work and development cost of database migration.

References

[1] Murugesan S. Understanding Web 2.0[J]. IT Professional,2007, 9(4): 34-41.
[2] Kubiatowicz J, Bindel D, Chen Czerwinski S. OceanStore：an architecture for global-scale persistent storage. SIGARCH Compute. Archit.News 28, 5(Dec.2000), p.190--201.

[3] Oreilly T. What is Web 2.0: design patterns and business models for the next generation of software[J]. International Journal of Digital Economics, 2007, 65(3): 17-37.
[4] Stonebraker M. SQL databases v. NoSQL databases[J]. Communications of the ACM, 2010, 53(4): 10-11.
[5] Chang F, Dean J, Ghemawat S, et al. Bigtable: a distributed storage system for structured data[C].Berkeley, CA, USA: USENIX Association, 2006: 15.
[6] Hadoop. Available at: http://hadoop.apache.org.