International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 4, Issue 6, November - December 2015 ISSN 2278-6856

CROSS DATABASE MANIPULATOR USING COMMON INTERFACE

Mr. Omkar Singh, Dr. S. K. Singh, Pranoti Hatkar,Tanvi Sawant,Siddhi Satam

Thakur College of Science and Commerce, Thakur Village, Kandivali – E, Mumbai – 400101

Abstract

Cross database manipulator using common interface is a generic database converter. This application migrate database structure, data, views and stored procedures from one database to another. This application helps the user to convert data from one database to another database. This application reduces the risks associated with the present system by providing easy way of accessing and handling a database. The special feature of this application is that by using this application the user can interact with multiple databases. It provides an easy way to use common interface for viewing, adding, editing, and deleting entries in almost all the databases.

Keywords- Database, manipulator, database converter, migration, common interface.

1. INTRODUCTION

1.1 MANIPULATOR

Cross Database Manipulator is an application which helps the user to convert different databases. This application is useful to migrate from one database to another. Using this application user can create, review, and modify migration definition with/without database connectivity. This application allows user to migrate database tables, views, data, synonym, stored procedure from one database to another.

Cross database manipulator is Cross-Platform database independent software. It provides an easy way to use common interface for viewing, creating, editing, or deleting tables, views and stored procedure in almost all the databases. This also allows user to insert, update, delete or alter records form database tables through SQL window. The user should never have to write SQL to view the data although a SQL window is provided. All user tables and basic data type fields are available for access in the selected database. Cross database manipulator provides a feature to modify data type, length and name of the column to be specified in the target database tables. User can select as many tables as he wants from source database to migrate into target database.

1.2 NEED OF MANIPULATOR

Database schemas have large data which is collected for years and it need to be upgraded to a new application, which uses different schemas. For doing that we need to convert data from source schema to the new schema. Cross-Database Converter is a tool focused on database migration. Besides solving general migration issues, it also automatically handles a great number of minor differences between databases by generating intelligible code. By ensuring a flexible migration process, it allows customize the conversion to fully meet needs. Although industry leading database engines like MySQL, MS SQL Server, Oracle, MS Access, PostgreSQL, SQLite, MS FoxPro and etc. are very similar to each other, but they are not identical in their supported data types, metadata organization or internal data manipulation capabilities. The manual data transfer process is a time-consuming and difficult task that is also prone to many errors at a destination database after migration.

Even if you're not an expert in database server administration, with a few clicks of the mouse, the data is converted to the database of your need. They are being used to accomplish data import and export in both forward and backward directions.

1.3 EXISTING MANIPULATORS

For the cross database manipulation, existing converter did conversion of single database or else we have to perform the conversion manually by entering the data in database of another technology. Or else we have to prepare the script for the conversion which can be made by the expertise in the script making; people not having depth knowledge about the subject w ere not able to perform the migration. Manual method of conversion was very time consuming and prone to errors. At the time of manual entry in the destination database we might enter wrong data type as of source database. This can create a problem for users. Also, the database with small size (less number of records) can be converted by manual method but big database (with greater number of records) is very difficult to convert by manual methodology. In the existing manipulator we have to convert single record from source database to the destination database we cannot convert the whole table in the existing manipulator. Stored Procedures, views, synonyms plays major role in the database; but converting them from source to destination database involves coding (knowledge) of them, in the existing manipulator.

1.4 IMPORTANCE OF DATABASE

Database management systems are important to businesses and organizations because they provide a highly efficient method for handling multiple types of data types. Some of the data that are easily managed with this type of system include: employee records, student information, payroll, accounting, project management, inventory and library books. These systems are built to be extremely versatile.

Without database management, tasks have to be done manually and take more time. Data can be categorized and structured as per the needs of the company or organization. Data is entered into the system and accessed on a routine basis by assigned users. Each user may have

an assigned password to gain access to their part of the system. Multiple users can use the system at the same time in different ways.

2. SYSTEM DESCRIPTION

This application is designed so that anyone can easily migrate their database from one to another. It provides fast conversion of databases. User can also modify table data using this application.

This application will provide following capabilities:

2.1 Table Manipulation

This feature provides a user with migration of tables. User is allowed to migrate as much number of tables as he/she wants. User can also can create or delete tables as per their choice.

2.2 Record Manipulation

Cross Database Manipulator provides a capability to manipulate table records such as add, alter and remove tuples.

2.3 Query Manipulation

This feature allows manipulation of various queries that takes place in two different databases queries having similar functionality might have different syntax. For this purpose SQL window is provided where user can perform any query.

2.4 Generation of logs

The application can generate verbose logs. This specifies the details of migration such as date and time of migration, number of selected tables, views, stored procedures for migration, status of migration.

2.5 Procedure Manipulation

This feature allows the conversion of different procedures that are used in different databases.

2.6 Size Manipulation

User can performs manipulation of data type size that is distinct for different databases.

2.7 Data Type Manipulation

Multiple databases can have different data types that have same function but different name; such data types can be converted.

For example: In MYSQL, “BIGINT” is used for numeric data whereas in Oracle “Number” data type is used.

2.8 View Manipulation:

For different databases different views are created hence to convert them from one database format to other, view manipulation is performed.

2.9 DATABASE DESCRIPTION

This application mainly used to convert four databases ie. Microsoft SQL Server, Oracle, IBM DB2, MYSQL. For Backend we are using Microsoft SQL Server 2014 as the default database.

2.9.1 Microsoft SQL Server

Microsoft SQL Server is a standard language for accessing databases. Microsoft SQL Server is a relational database management system developed by Microsoft. It is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).[1]

In addition to Microsoft SQL Server 2014 we are also using three other database languages they are:

2.9.2 Oracle

Oracle Database is an object relational database management system produced and marketed by Oracle Corporation. An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

2.9.3 IBM DB2

IBM DB2 is a family of database server products developed by IBM. These products all support the relational model, but in recent years some products have been extended to support object-relational features and non-relational structures, in particular XML. DB2 is a family of relational database management system (RDBMS) products from IBM that serve a number of different operating system platforms. It is a full-featured, high-performance database engine capable of handling large quantities of data and concurrently serving many users.DB2 can be administered either through a command-line prompt or a GUI.[2]

2.9.4 MySQL

MySQL is the world's second most widely used open-source relational database management system (RDBMS). MySQL is a popular choice of database for use in web applications. Free-software-open source projects that require a full-featured database management system often use MySQL. MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.[3]

3. MAPPING

3.1 FUNCTIONS

3.1.1 CONCATENATION

Concatenating two strings is done with the || operator: string1 || string2

Standard	If at least one operand is NULL, then the result is NULL. It's unclear to me if the DBMS is allowed to try to automatically cast the operands to concatenation-compatible types.
MSSQL	Does not automatically cast operands to compatible types. If an operand is NULL, then the result is NULL. Badly breaks the standard by redefining || to mean OR.

Oracle	Automatically casts values into types which can be concatenated. As Oracle interprets NULL as the empty string, it doesn't return NULL if an operand is NULL.
IBM DB2	Does not automatically cast concatenated values into compatible types. Breaks the standard by using the '+' operator instead of '‖'.
MySQL	Automatically casts values into types which can be concatenated. If an operand is NULL, then the result is NULL. As Oracle interprets NULL as the empty string, it doesn't return NULL if an operand is NULL.

3.1.2 LOCALTIMESTAMP

It's often important to get the value of current date and time. Below are the functions used to do that in the different implementations.

Standard	The current timestamp (without time zone) is retrieved with the LOCALTIMESTAMP function which may be used as: SELECT LOCALTIMESTAMP ... or SELECT LOCALTIMESTAMP(*precision*) ... Note that "SELECT LOCALTIMESTAMP() ..." is illegal: If you don't care about the precision, then you must not use any parenthesis. If the DBMS supports the non-core time zone features, then it must also provide the functions CURRENT_TIMESTAMP and CURRENT_TIMESTAMP(*precision*) which return a value of type TIMESTAMP WITH TIME ZONE. If it doesn't support time zones, then the DBMS *must not* provide a CURRENT_TIMESTAMP function.
MSSQL	Doesn't have the LOCALTIMESTAMP function. Instead, it has CURRENT_TIMESTAMP which however doesn't return a value of TIMESTAMP WITH TIME ZONE, but rather a value of MSSQL's DATETIME type (which doesn't contain time zone information).
Oracle	Follows the standard.
IBM DB2	Doesn't have the LOCALTIMESTAMP function. Instead, it provides a special, magic value ('special register' in IBM language), *CURRENT_TIMESTAMP* (alias to 'CURRENT TIMESTAMP') which may be used as though it were a function without arguments. However, since DB2 doesn't provide TIMESTAMP WITH TIME ZONE support, the availability of CURRENT_TIMESTAMP could be said to be against the standard—at least confusing.
MySQL	Follows the standard.[4]

3.2 DATA TYPES

3.2.1 SQL Server to Oracle

Index	SQL Server	Oracle
1	bigint	integer
2	binary	raw. (If precision is greater than 2000 bytes, then it is converted as BLOB)
3	bit	number(1)
4	char	char
5	datetime	date. (If the Oracle version is 9i or above, datetime is converted as timestamp)
6	decimal	decimal
7	float	float
8	image	blob
9	int	int
10	money	decimal (19, 4)
11	nchar	nchar
12	ntext	nclob
13	numeric	numeric
14	nvarchar	nvarchar2. (If the precsion is greater than or equal to 2000 bytes, it is restricted to 2000)
15	real	real
16	smalldatetime	date. (If the Oracle version is 9i or above, datetime is converted as timestamp)
17	smallint	smallint
18	smallmoney	decimal (10, 4)
19	timestamp	raw(8)
20	tinyint	smallint
21	text	clob
22	varbinary	raw. (If size is above 2000 bytes, it is converted as BLOB)
23	varchar	varchar2. (If size is greater than 4000 bytes, it is converted as CLOB)

3.2.2 SQL Server to MySQL

Index	SQL Server	MySQL
1	bigint	bigint
2	binary	longblob
3	bit	tinyint
4	char	char. (If size of char is greater than 255, then it is mapped to text)
5	datetime	datetime
6	decimal	decimal
7	float	float
8	image	longblob
9	int	int
10	money	decimal (19, 4)
11	nchar	national char. (If size of nchar is greater than 255, then it is mapped to text)
12	ntext	longtext
13	numeric	numeric
14	nvarchar	national varchar. (If size of nvarchar is greater than 255, then it is mapped to text)
15	real	real
16	smalldatetime	datetime
17	smallint	smallint
18	smallmoney	decimal (10, 4)
19	timestamp	timestamp
20	tinyint	tinyint
21	text	longtext
22	varbinary	longblob
23	varchar	varchar. (If size of varchar is greater than 255, then it is mapped to text)

3.2.3 Oracle to SQL Server

Index	Oracle	SQL Server
1	char	char
2	nchar	nchar
3	varchar, varchar2	varchar
4	nvarchar2	nvarchar
5	number	numeric
6	integer	int
7	date	datetime
9	long raw	image
10	blob	image
11	clob	text
12	nclob	text
13	rowid	char(18)
14	bfile	image

3.2.4 Oracle to DB2

Index	Oracle	DB2
1	char	char
2	nchar	char
3	varchar, varchar2	varchar
4	nvarchar2	varchar
5	number	numeric
6	integer	integer
7	date	timestamp
8	long	clob
9	raw	blob(2000)
10	long raw	blob
11	blob	blob
12	clob	clob
13	nclob	clob
14	rowid	char(16) for bit data
15	bfile	blob

3.2.5 Oracle to MySQL

Index	Oracle	MySQL
1	char	char. (If size of char is greater than 255, then it is mapped to text)
2	nchar	national char. (If size of nchar is greater than 255, then it is mapped to text)
3	varchar, varchar2	varchar. (If size of varchar is greater than 255, then it is mapped to text)
4	nvarchar2	national varchar. (If size of nvarchar2 is greater than 255, then it is mapped to text)
5	number	numeric
6	integer	integer
7	date	datetime
9	long raw	longblob
10	blob	longblob
11	clob	longtext
12	nclob	longtext
14	bfile	longblob

3.2.6 DB2 to SQL Server

Index	DB2	SQL Server
1	char	char
2	varchar	varchar
3	decimal	decimal
4	double	float
5	integer	bigint
6	real	real
7	bigint	bigint
8	smallint	smallint
9	timestamp	datetime
10	time	datetime
11	date	datetime
12	blob	image
13	clob	text
14	dbclob	ntext

3.2.7 DB2 to Oracle

Index	DB2	Oracle
1	char	char
2	varchar	varchar2
3	decimal	decimal
4	double	double precision
5	integer	integer
6	real	real
7	bigint	integer
8	smallint	smallint
9	timestamp	timestamp
10	time	timestamp
11	date	date
12	blob	blob
13	clob	clob
14	dbclob	nclob

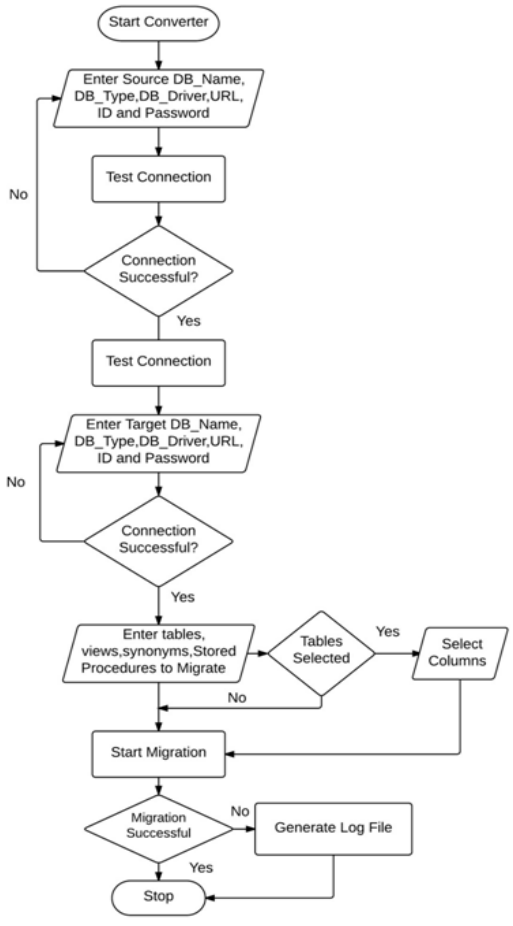
3.2.8 MySQL to Oracle

Index	MySQL	Oracle
1	tinyint	smallint
2	smallint	smallint
3	mediumint	integer
4	int, integer	int
5	bigint	integer
6	datetime	date
7	date	date
8	timestamp	date
9	year	n/a
10	char	char
11	varchar	varchar
12	binary	raw. (If the size of binary is greater than 2000, then it is mapped to blob)
13	varbinary	raw. (If the size of varbinary is greater than 2000, then it is mapped to blob)
14	tinyblob	raw(255)
15	blob	long raw
15	mediumblob	long raw
16	longblob	blob
17	tinytext	varchar(255)
18	text	long
19	mediumtext	long
20	longtext	clob
21	enum	n/a
22	set	n/a
23	float	float
24	double	double precision
25	real	real
26	decimal	decimal
27	numeric	numeric

3.2.9 MySQL to SQL Server

Index	MySQL	SQL Server
1	tinyint	tinyint
2	smallint	smallint
3	mediumint	int
4	int, integer	int
5	bigint	bigint
6	datetime	datetime
7	date	datetime
8	timestamp	datetime
9	year	n/a
10	char	char
11	varchar	varchar
12	binary	binary
13	varbinary	varbinary
14	tinyblob	image
15	blob	image
15	mediumblob	image
16	longblob	image
17	tinytext	char
18	text	text
19	mediumtext	text
20	longtext	text
21	enum	n/a
22	set	n/a
23	float	float
24	double	float(53)
25	real	real
26	decimal	double
27	numeric	numeric

3.3 FLOW DIAGRAM



3.4 FUTURE ENHANCEMENT

In the proposed system we have used 4 database languages i.e. Microsoft SQL Server, Oracle, IBM DB2 and MySQL. In future more database languages can be implemented as per the user's requirements and new emerging technologies. It is also possible to implement more functionality such as permission, roles, users, groups etc. Mailing the migration log files to the user can also be added in future.

4. CONCLUSION

Data migration is a routine part of IT operations in today's business environment. Even so, it often causes major disruptions as a result of downtime or application performance problems, and it can severely impact budgets. To prevent these problems, organizations need a consistent and reliable methodology that enables them to plan, design, migrate and validate the migration. Further, they need migration procedures that support their specific migration requirements, including operating systems, storage platforms and performance. In addition, migration products that maintain continuous data availability during the migration without affecting performance are desirable. The migration procedures mentioned in this document provide a server-side infrastructure and new high-speed, parallel Export and Import utilities for highly efficient bulk data and metadata movement between databases. We can now move data and metadata between databases between different platforms faster and easier than ever before, and this methodology helps you meet your various organization needs.

5. ACKNOWLEDGMENT

We have taken efforts in building this software. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them.

We would like to express our gratitude towards our parents & all the people who have helped us in many different ways possible. And also by giving encouragement which help us in completion of this project.

We would like to express our deepest appreciation to all those who provided us the possibility to complete this project.

REFERENCES

[1]. https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CB0QFjAA&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FMicrosoft_SQL_Server&ei=JEyKVeixN4aeugSXzJD4AQ&usg=AFQjCNHXKyKbK5gH3pGfa_Ys2wVgFUF2Rg&bvm=bv.96339352,d.c2E [24 May,2015]

[2]. https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0CCMQFjAB&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FIBM_DB2&ei=-E2KVdPrEMTnuQT-wICIDg&usg=AFQjCNHpD3BhtSwyv-c94phK1yU_kuymIA&bvm=bv.96339352,d.c2E [28 May,2015]

[3]. https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&cad=rja&uact=8&sqi=2&ved=0CDMQFjAD&url=https%3A%2F%2Fen.wikipedia.org%2F%3Ftitle%3DMySQL&ei=ME-KVaz9As6PuASi7J-ABg&usg=AFQjCNF5Rvl2sVjnOqGNvrO-SUOGiFC9tw&bvm=bv.96339352,d.c2E [29 May,2015]

[4]. https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&sqi=2&ved=0CB0QFjAA&url=http%3A%2F%2Ftroels.arvin.dk%2Fdb%2Frdbms%2F&ei=7EaKVfWiL4OQuATXmpCICg&usg=AFQjCNE8NVVvGx9yt4UIg_V-s8QbSTGvbg&bvm=bv.96339352,d.c2E [29 May,2015]