Data Migration Automation Tool

Prof. Saleem Durai M.A. , Anirban Mishra, Garvit Khandelwal
#1 Associate Professor (Computer Science) Vellore Institute of Technology, Vellore Tamil Nadu, 9443686556
#2 Btech 4th Year(Computer Science) Vellore Institute of Technology, Vellore, Tamil Nadu, 7639473294
#3 Btech 4th Year(Computer Science) Vellore Institute of Technology, Vellore, Tamil Nadu, 9566812795

Abstract

In the present age, the information age each and every day we get loads of data which needs to be stored in a database for further processing. Database technology is one of the fundamental requirement for building a basic information system. Most companies not only use one but several database system . Some of which may be network or relational, hierarchical and many now-a-days are using the object oriented database. Each and every day new technologies are invented due to which the companies are ready to transfer to the new technologies. Sometimes while transferring from one technology to another there we need a transfer from one database system to another. The transfer of data from one database to another is a quiet tedious task which requires more work force and time to time quality check. There are are many design constraints like the new system should be able to handle all the data formats that were present in the earlier database. A good data migration tool is required to take care of all the constraints and transfer the data swiftly. The tool would take care of three stages in the process. I have categorized the data transfer process into three stages. This tool would transfer the data in minimum time, minimum cost less work force and good data quality. Although the transition from one database system generation to the next is important, there exists only a few migration strategies that exists. In the database migration tool that I have implemented, I used a migration strategy which is described later in the paper . Among all the companies the telecoms industry are one of the largest users of database. Each company has millions of customer. In India itself there is an addition of 7 million new customer at an average each year. Their various details like name, address, sex, rate tree, voucher details needs to be stored in a database. As most of the companies are more than 10 years old, they are using SYBASE database. Now most of them need to update their database to ORACLE. This is because of various features provided by the Oracle database like Oracle databases perform each transaction in isolation from others, which ensure higher level of security, once all the transactions are successfully completed, the results are to be made permanent and survive media failures, the latest versions of Oracle databases have a recycle bin option (This works exactly like the Microsoft Recycling Bin), while MySQL may flutter when the data grows in size, but Oracle performs just normally as it would have worked earlier. So in this paper I would basically be concentrating on transferring data from a SYBASE database to an Oracle database.

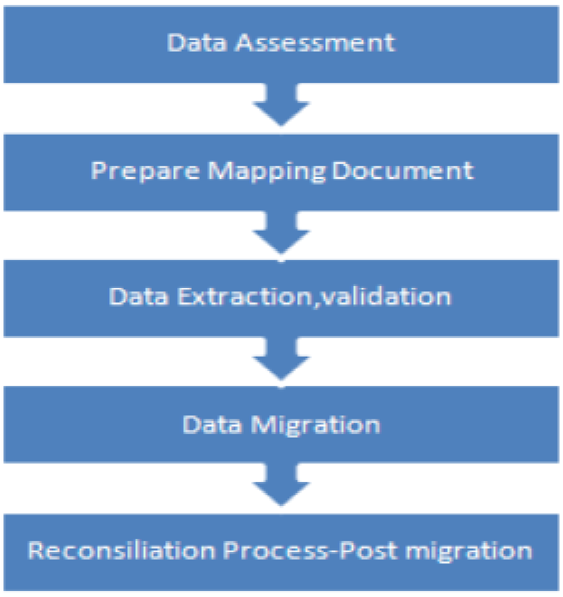
Introduction

Data migration may appear as a simple concept but from the industry experience we can see that it is one of the most risky, highly time consuming and the most complex project in the IT industry. The main reason is that the business performance will suffer if the data does not get migrated properly or if any ambiguity in data arises. Some of the other problems can be because of redundant data it might consume large amount of space then required. The main things that must be kept in mind while designing an automation tool are as follows:-

i. First of all the cost that is involved in data migration must be reduced.
ii. Secondly, we must find an optimized way so that the time taken by the tool is minimum.
iii. Thirdly, the data quality must be good. By data quality here I mean the the data must be stable, there should be no redundant data, supports all the business functionality that it used to support in the earlier database.
iv. Lastly the most important it should be able to maximize the business value of the data. Here by business value I mean that the base reason for which the database migration was initiated must be preserved. Suppose we are migrating to Oracle database then all the functionality that are provided by the database providers must be usable on the data that was migrated because this was the basic reason due to which we thought of going for migration. The main objective of this paper is to provide the user an idea of what are the basic and fundamental functionality required to make a database migration automation tool. It would contain all the steps involved like validation of data, pre and post migration tasks and the most important the migration process itself. This paper will help Quality Analyst, Database administrator, Software Developer and others who are mainly involved in the migration of the data from one platform to another as well as across domains.

Methodology

The various stages in the process of data migration can be known from the diagram below.


Data Assessment
Prepare Mapping Document
Data Extraction,validation
Data Migration
Reconsiliation Process-Post migration

The way to implement each and every step will be discussed in the implementation step. Here I would discuss in detail about what would be happening in the stages that are described in the figure above.

➔ The data assessment stage is the stage where we try to learn about the data and the database. Here by learning means we need to analyze the tables, the primary key and how various tables are inter related through the foreign key. We need to learn the various rules governing both the databases and the validations required for various fields like NULL or Unique etc.

➔ In the next stage we prepare the mapping document. The mapping document is a document in which there would be a mapping of the columns of the the old and the new database. It would also contain the validation rule governing each columns. It is a manual task and is an one time task. Once this is prepared it can be used to transfer millions of data without having to modify this document.

➔ The data extraction and validation phase would extract the data from the old database and store it in a suitable format so that we will be able to transfer it easily and in less time. I have used to store it in a .dat file and the values are stored in it separated by commas.

➔ The data migration process stage is the main core process of our paper. The real data migration takes place in this stage. I have further divided this stage into three stages namely the pre migration, migration and the post migration task. These stages will be discussed in detail under the topic implementation.

➔ The last stage would involve the process after the migration task was carried out. Here it would involve the tasks like saving the database if the migration was successful, or else rolling back the state of the database if the migration was unsuccessful, deleting the file that were created for the migration of a particular batch, deleting the data in the temporary database etc.

Implementation

Pre Migration Task

Before starting with the migration process one must know about the source and target database. The tables present in the source database and what are the tables in the target database. What is the amount of data to be transferred is one of the important aspect of data migration. We must know all the relations that exist among the tables. Basically we must have a clear picture of the whole database design of both the source and target database. Then one must be clear about the various rules governing each database and the rules followed by the company to store various data. For example for marital status a company might use “MARRIED” as a value while other company might use just “M” as the value. So these constraints must be clear. Then the first step would be to define the range of data we are going to migrate. On the basis of the primary key (or one may use some other strategy) we can decide the range for which we want to transfer the data from one database to another .We can make a list of the table names to be migrated and store it in a file. Now we have to create .dat files for each tables. These .dat files will be used to store the data that we retrieve from the source table. I call these files as .dat files because these files belong to database so the short form dat.

Once the range of data is decided we can write a script to retrieve the data from the table and I have used it to store these data in comma separated format in the .dat file that we had created for the table earlier. We are using this process because in my case the server for the target database was different from the source database. So in order to make the transfer of data quicker from one database to another we will be moving these dat file. Now once these data files are ready we can concentrate on the mapping of the source to target database.

The mapping of the source database to target database is one of the important process to achieve good quality data after the migration process. Now for mapping we can create .cfg files. I call this as .cfg files because it is the short form of configuration and this file is used for configuring or act as an interface between both databases. Basically one must know the correct corresponding columns in each of the source and target database. For example a

column in one database may be “ADDR_STREET_NUMBER” while in the target database it may be referred as “ClientPersonalDetails.Address1”. So one must know these mappings between the databases. There are various data migration tools available in the market now-a-days. But most of them does not allow the change in schema of the database (like changing the column name as in the case above). They just transfer the database to the target database by maintaining the schema of the source database in the target database. Once we know all the mappings we can put them into the cfg files. Here entering into the cfg files one can follow any format. For example one may go with:-
ENTRY_DATE=SIGNUP_DATE
or one can use the following which is more elaborate in nature and will be easier to use during the further programming:-

Schema	Table	Column	Schema	Table	Column	Validation Rule
Target	Customer	Cust_no	Source	Emp	Emp_no	Field is mandatory

If the mapping is done in the above format the coding afterwards will be easier because from this we can clearly know from which database and which column it is to be transferred and to which column in the new database it is to be inserted.

Now that we are ready with the .dat files and the .cfg files there remains one final step before going on with the migration procedure. This step is to check if the data extracted from the source database is compatible to be inserted into the target database. By compatible I mean that the rules governing the source database are usually different from the target database. For example the database that I used as source was SYBASE while the target database was ORACLE so the basic things that I had to check were:-
User names were exported to .dat file as existing in SYBASE, but converted to upper case (company's requirement). This approach might require manual action because some user names might not match Oracle constraints. To ease the possible task of adjusting the mentioned files, we can write a script with the above functionality. The above script can be used after creating the batch. Here by batch I mean the group of record that we chosen to undergo the migration process. It is recommended to run this script after creating the batch , update the files according to check script output and rerun the script to double check if the problematic cases have been solved. This can be repeated until no errors are found anymore. In my case the groups of user names which have to be adjusted manually into four categories:

Special character check

SYBASE allows special characters and blanks in user names. This category will list those SYBASE user names which contain other characters than [A-Z] or [0-9].

Length check

The source server and SYBASE allowed user names with a length up to 20 characters. This category will list those SYBASE user names which exceed a length of 16 characters.

Duplicate check

SYBASE allows lower case user names. This category will list those SYBASE user names which are not unique. This might be caused by converting them to upper case or deleting special characters from user name or cutting user name with length > 16 characters.
Similarly according to ones requirement one can prepare a script to make the data ready for the migration process by checking the special condition which might lead to an error if transfer process is started and the target database doesn't support the data of that type. If it occurs then we have to delete all data transferred and restart the process again which would take a lot of time. For this also there is a quicker method to roll back to the original state of the database which is easier and quicker than manually deleting and rolling back. I will discuss that process after the whole migration process is completed.

Till now whatever we have discussed I would like to categorize them as pre migration tasks because these task do no migrate the data. They just help in preparing the data for the transfer process.

The Migration Process

In the migration process I would like to define it as a three stem process.

1. Loading of data into a temporary database.
2. Transferring data into target database.
3. Checking and releasing data into that database.

In the first step we must create a .dtp file short form for database transfer file. This file would contain the order in which the tables will be loaded into the target database. This file is highly required because most of the companies have tables in which the data is related. For example for a company storing employee details will have an employee id as a field. This field will be acting as foreign key to all other tables. So if the order gets reversed then the data will not be inserted because it will display an error that parent key not found.
Once the .dtp file is created the order of table insertion will be defined and there would be no error while inserting. Next we can begin the loading of data into the temporary database. Here we are using a temporary database because if the process fails then the index of the database may become unstable and we don't want this to be happening to our original database. So during the load process if anything goes wrong then we can just flush the data and then restart the process again after fixing the error. To do this loading we can write a script which would do the loading process in the following manner. It should first read the .dtp file in order to know which table to load first. Then I must read through the .cfg file in order to get to which column we must insert the data and the other validation rules available in it. Lastly it must find the .dat file the data of which is to be loaded into the database. Once it has all the data available for a record then it can use them to insert into the database. This process can be carried out recursively so that all the data gets transferred into the database. A log file can be maintained so that any error message can be thrown into it. Not only error message each and every query that is being executed must be written to these log files so that we can know where the error occurred.

In the second step we should check the log files created in the first step to see if there was any error. If any error existed then we must restart the process again. If there is no error and the process is successful then we can confirm that the database was transferred properly. No we can start the process of transferring the data from the temporary database to the permanent database. This process does not require any of the above processes because it already in the

Oracle temporary database we just have to move it into permanent database. As in the previous case we can maintain a log file in this case also. This process might create an error if any required field has a null value.

In the last step again we have to check the log files if the transfer was completed successfully. If there was some error then we can use the concept of flash to roll back instead of deleting all the data manually which would take a lot of time. The concept of flash will be discussed later. If there are no errors then we can save the database (commit the database) and then release the connection. This process is important because this process would save the database and once this process is executed then the database cannot be rolled back into the previous state.

This whole process I name it as migration task because this is the core part of the coding and is fully involved in the migration task.

Post Migration Task

All the above process must be carried out by disabling the GUI (Graphical User Interface) the uses the background database for its execution. Because if the GUI is running simultaneously then it might lead to the instability of data because it might change some of the value that are in the process of migration. So after the migration is successful we must restart the GUI and check if it is behaving normally. This is required to be done because sometimes due to dissimilar query syntax the GUI may behave abnormally. So these conversions in syntax must be done manually.
After the migration process the data from the temporary database must be flushed out and the database must be made stable for the transfer of the next batch.

Everywhere during the migration process I have used the word batch. This is because the above process can migrate the data as a whole but it is normally advisable because when the database is huge the server performance might be reduced and it might take a lot of time. The other problem is that it is possible for human error if the database used is huge. If there is some human error then all the labor done to transfer would go in vain. So it is advisable to transfer the data in small batches for easy, faster and would produce data of higher quality.

Earlier I had mentioned about flash that I am now going to elaborate on. It is nothing but a simple concept provided by most database providers which helps in quick data recovery in case of any errors. The basic concept behind it is that every time before doing any changes to the database we are required to take an image of the original database. After taking the image we can go with the transfer process. If something goes wrong in the transfer process then we can roll back to the original database by just deleting the table as a whole and then loading the image of the database that we had taken earlier. While transferring millions of data I found that this feature of flashback takes very less time than manually doing the whole rollback. This is because it is easier to delete a table as a whole than deleting some of its data. Since deleting some of the data would take a lot of comparisons before actually deleting the data. While in the case of dropping the table it would make only one comparison i.e. it will check only for the table name. From experience it is found that deleting of data takes more time than rather inserting it. So to save the time to roll back it is better to drop the table and then load the earlier stable database image. This feature is provided to the Oracle database users and most of other database providers have also included properties similar to flash in Oracle.

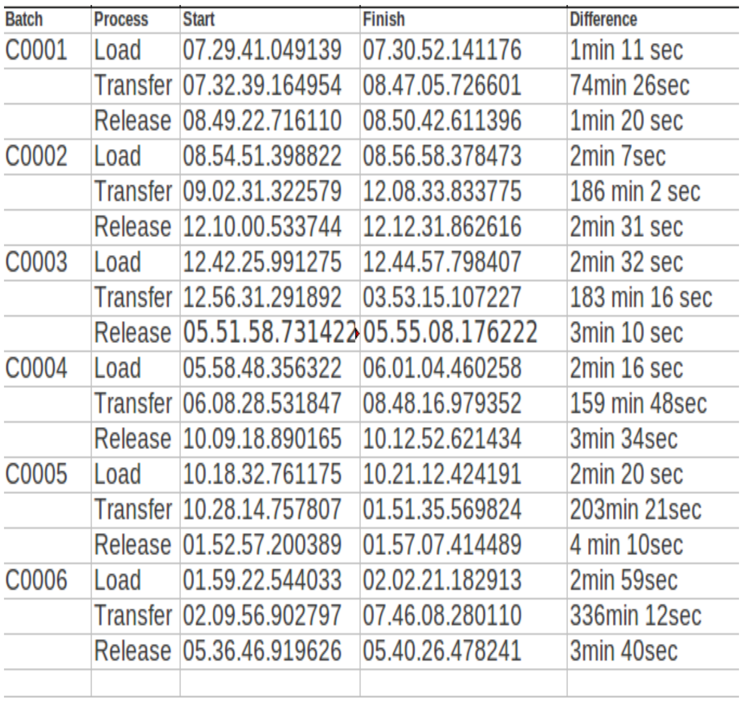
Result

The tool is able to migrate data from SYBASE to (ORACLE) successfully through the 3 stages the pre migration task where we set all the values that are essential for migration, the migration process where the migration actually takes place and then the post migration process which includes activities that makes the final changes in the database and releases all the resources. The process creates an intermediate database schema into which it moves the data. The reason of this intermediate database has already been described above. It supports progress of the batch. Here by progress we mean that how far the transfer is completed . Databases can be flashed back at any point in case something goes wrong.

Here I have captured the result on the real world data that were provided to me by my company. The benefits of this tool were found to be

- Speed
- Accuracy
- Timeliness
- Precision
- Other quality factors

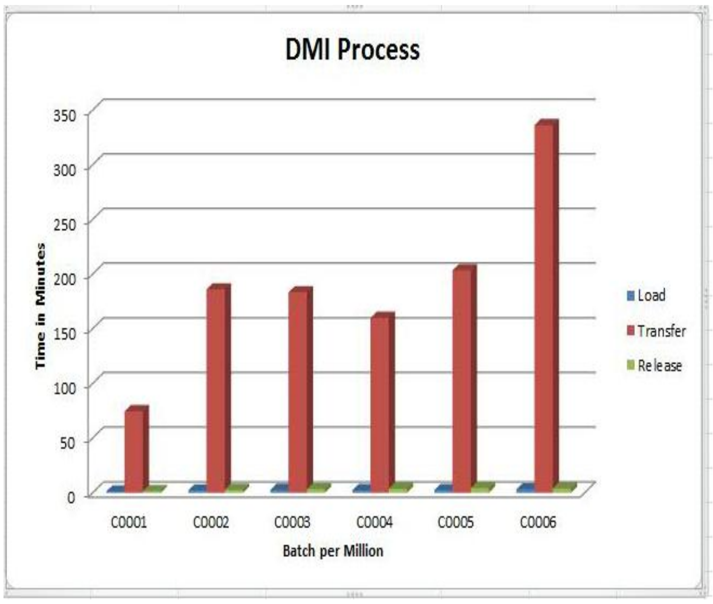
We had data for about 6 million. We created 6 batches containing one million data each. Then we ran all the three processes on each batch at a time. The below table gives the start time, end time and the time taken to execute each process for a batch. From the table it is clear that batch took around 3 hours to be transferred from SYBASE to Oracle. With the help of more optimization and the use of good servers this time can be brought down to around one and half hours.



Batch	Process	Start	Finish	Difference
C0001	Load	07.29.41.049139	07.30.52.141176	1min 11 sec
	Transfer	07.32.39.164954	08.47.05.726601	74min 26sec
	Release	08.49.22.716110	08.50.42.611396	1min 20 sec
C0002	Load	08.54.51.398822	08.56.58.378473	2min 7sec
	Transfer	09.02.31.322579	12.08.33.833775	186 min 2 sec
	Release	12.10.00.533744	12.12.31.862616	2min 31 sec
C0003	Load	12.42.25.991275	12.44.57.798407	2min 32 sec
	Transfer	12.56.31.291892	03.53.15.107227	183 min 16 sec
	Release	05.51.58.731422	05.55.08.176222	3min 10 sec
C0004	Load	05.58.48.356322	06.01.04.460258	2min 16 sec
	Transfer	06.08.28.531847	08.48.16.979352	159 min 48sec
	Release	10.09.18.890165	10.12.52.621434	3min 34sec
C0005	Load	10.18.32.761175	10.21.12.424191	2min 20 sec
	Transfer	10.28.14.757807	01.51.35.569824	203min 21sec
	Release	01.52.57.200389	01.57.07.414489	4 min 10sec
C0006	Load	01.59.22.544033	02.02.21.182913	2min 59sec
	Transfer	02.09.56.902797	07.46.08.280110	336min 12sec
	Release	05.36.46.919626	05.40.26.478241	3min 40sec

The data was then successfully validated after the migration process. The number of record in the older database was found to be the same as the newer database. From the above table it can be seen that the last process took more than 5 and half hours, this was because the connection between the server was lost in the middle of the process so we had to flash back the database and then restart the process.

Below is the graph showing the details of the three process that were executed for each batch.



Conclusion

The tool successfully automates the process of Data Migration between SYBASE and ORACLE database. The designed tool would work for any number of subscriber data. This model is working fine with main frame source data migration into Oracle database. This process of migration where huge amount of data is to be migrated between databases in less or more accurate and reducing effort by 40% to 50%. The benefit of this tool is saving of time, money, and data quality will be high around 30% when compared to manual process and its 21% efficient with respect to time and defects raised.

Future Enhancement

In future, the tool would also be able to add functionality for displaying a progress bar. The tool would also be able to migrate from other databases. A GUI can also be developed over the present framework in case it's needed. Global design of a corporate – wide data architecture will let this tool to be used by other companies as well.

References

[1] Bloor Research (2007)

[2] Andreas, Meier, CSSInsuran -Providing Database Migration Tools

[3] Ravindra, S. Hegadi, Mohan H S-Automated Data Validation for Data Migration Security

[4] Rodgers J.: Database Coexistence -Requirements and Strategies

[5] There were other research material that were internal to my company(Ericsson) which cannot be revealed publicly since these are the internal materials allowed only for the review by Ericsson employees.