DATA MIGRATION ACROSS THE CLOUDS

Ashutosh Dhar Dwivedi

Amity School of Computer Science, Noida (India)

ABSTRACT

Before few years back one-off batch cleansing jobs were used for transferring or moving data from one computer system to another, which was complex job for everyone. But now cloud offers convenient way to connect all system together. Having an efficient strategy for optimizing long distance, data migration is essential and required for every data center. It means it is easier to manage data across various different customer sites as cloud bridges the gap between different systems.

Index Terms -- Cloud Computing Services, Data Migration Technical Issues, Application Migration, Data Migration

Data migration with low cost of transmission is challenge for the current environment. Moving data across different geographical domains with less number of steps and having less transmission time is very much important for current situation.

This paper will explore the issues and method of Data Migration with better way in terms of cost, transmission time and number of steps across the Clouds.

I. INTRODUCTION

Cloud computing

Cloud computing describes storing and accessing data and programs over the internet instead of your computer's hard drive. What cloud computing is not about is your hard drive. While if data is stored on the hard drive that is called local storage. Cloud computing entrusts remote services with a user's data, software and computation as shown in figure1.1.

Fig 1.1:Cloud Computing

1.1 Cloud Computing Services

Main Cloud Computing Services are given below:

a. Infrastructure-AS-A-Service.

b. Platform-AS-A-Service.

c. Software-AS-A-Service.

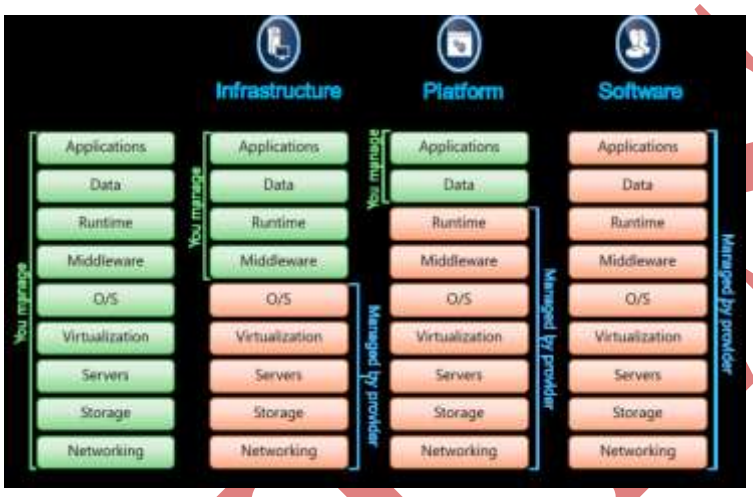

Fig1.2:Cloud Computing Service Model

1.1.1 Software as a Service (SaaS)

Short for **S**oftware **a**s **a** **S**ervice, SaaS is a software delivery method that provides access to software and its functions remotely as a Web-based service. There are many examples of SaaS vendors Salesforce.com, Google Apps, Ning, Cenzic, etc.

1.1.2 Platform as a Service (PaaS)

(PaaS) Platform as a Service is a delivery of a computing platform over the web. PaaS enables you to create web applications quickly, without the cost and complexity of buying and managing the underlying software/hardware. Some examples of PaaS vendors include Microsoft Azure, Amazon, Force.com

1.1.3 Infrastructure as a Service (IaaS)

(IaaS)Infrastructure as a Service is a provision model in which an organization outsources the equipment used to support operations, including storage, hardware, servers and networking components. Some examples of IaaS vendors include Amazon, Rackspace, CloudFoundry.

II. DATA MIGRATION ACROSS THE CLOUDS (TECHNICAL ISSUES)

The amount of time it takes to complete the actual migration of objects and data from one database is relatively less than the amount of time it takes to complete an overall migration from assessment to production rollout. Migrations of one relational database to another are comparatively easier than migrations of a non- relational database to a relational database, because the organization of objects in a relational database is quite similar

compared to non-relational databases such as hierarchical and network databases. All major relational database vendors also offer tools that provide robust migration capabilities in an automated fashion. Regardless of the level of automation and success factor of any migration tool, however, sometimes manual intervention will be required when migrating from one database to another. Database migration tasks can be divided into the following categories:

- **Database schema migration**
- **Data migration**
- **Database stored program migration**
- **Application migration**
- **Database administration script migration**

Of all the migration tasks listed, the application migration task requires the most manual effort, although new tools and technologies are being developed to facilitate this task.

2.1 Database Schema Migration:

Database schema migration essentially involves migration tables, indexes, and views in a database. Relational databases are similar in terms of how their data is organized in tables and indexes, but they are different in terms of additional extensions to these tables and indexes that are designed to improve performance and facilitate development. Most migration tools can convert the database schema relatively quickly and accurately. Target-specific database schemas can also be generated from modelling tools such as Erwin. These are the most important things to consider during database schema migration:

2.1.1 Ensuring completeness of the schema:

It is necessary to ensure that all objects from the source database have been migrated over to the target database. It is very common to have multiple schemas and databases to support an application. Having circular dependencies among multiple schemas and databases may result in errors during schema creation on the target database, as some of these dependencies may not exist when a particular schema is being migrated. After creating all the schemas in Oracle, all the objects that are marked as invalid need to be recompiled and verified to ensure that they are migrated successfully.

2.1.2 Tables with system functions as DEFAULT value clauses on columns:

Many databases support having system functions as the DEFAULT value clauses on table columns. In almost all cases, these system functions do not exist in the Oracle database. As a result, some tables may not be created in Oracle, making other dependent objects invalid. It is recommended that you analyze the log resulting from the schema creation task, and isolate and rectify such errors.

2.1.3 Using clustered indexes:

Clustered indexes in databases such as Sybase allow data storage in a physically sorted fashion to match the logical order (index). As data is added, it is sorted and stored in the order defined by the clustered index. This

helps to reduce the time it takes to return the sorted data and to retrieve data by co-locating the index as well as the actual data in the same object. The Oracle database provides similar functionality with index-organized tables (IOTs). In IOTs, the primary key columns and the non-key data are stored in the same object. This helps users avoid having to look up data in tables separately, after index lookups, while executing a query in Oracle.

2.1.4 Creating database users and role assignment: Proper database roles and privileges on objects must be assigned to users. Schema and object-level privileges can be grouped into roles and assigned to users as needed. Creating roles and granting them to users can help in managing many object-level privileges.

2.1.5 Changing object names:

Any changes to the database object names due to restrictions in the database, as discussed in the "Analysis and Design" section of this chapter, need to be identified and shared with all team members so that they can make suitable changes in their applications or other database components.

2.1.6 Partitioning database tables:

Oracle allows large tables to be partitioned into smaller segments for management ease and for better performance due to the database query optimizer's ability to prune partitions during query execution, resulting in a reduction in the overall amount of data scanned. Based on data volume, performance, and manageability requirements, some tables may be chosen for partitioning. Although many relational databases support table partitioning, they implement this feature differently in terms of the methods allowed for partitioning, such as range, hash, and composite partitioning.

2.2 Data Migration:

After database schema migration, some representative data from the source database is migrated to the target database to enable testing and to ensure that the data migration scripts or tools chosen for the task are configured properly. The most common approach for data migration is undoubtedly the use of scripts that execute database utilities to export data from the source database and import it into the target database (Oracle), because they are easy to use and are free.

Regardless of the tools and scripts used to perform data migration, migrations of very large databases require planning. When migrate very large databases (those with at least a few terabytes of data) it is important to have the right data migration strategy, have the appropriate tools, and, most importantly, use appropriate database features such as partitioning and compression. Migration of large databases is fraught with challenges, among them a narrow window of time and lack of system resources (e.g., staging areas for data files). The following data extraction and loading strategies can optimize the data extraction, transfer, and loading processes:

- Parallel extraction of data from the source database
- Loading of data into the target database in parallel
- Using multithreaded processes for data loading

- Avoidance of index maintenance during the data loading process
- Reduction of I/O operations and use of staging areas via named pipes for data transfer between source and target databases

2.3 Database Stored Program Migration:

The task of migrating database stored programs includes migration of stored procedures, triggers, and views which, in many relational databases, are used for implementing critical business logic. In databases such as Microsoft SQL Server and Sybase, stored procedures and triggers are used extensively by developers to support simple functions (e.g. the CRUD operations CREATE, READ, UPDATE, and DELETE). However, using stored procedures exclusively for CRUD operations can result in inflexibility because the type of operation executed against a table is limited by the functionality implemented in the stored procedure.

Major tasks associated with stored program migration are:

2.3.1 Cleaning and optimizing code:

Oracle SQL Developer and other migration tools support migration of stored programs very well. However, it is recommended that you test these converted stored procedures and triggers for accuracy and efficiency of the converted code. Developers can implement a simple business requirement in many ways, making it harder for tools to optimize all such coding techniques in the converted code. Stored procedures and functions with hundreds of lines of code or more should be verified and tested for efficiency in terms of database feature usage as well as optimized coding practices.

2.3.2 Handling errors in stored procedures and triggers:

For applications that depend heavily on stored procedures and triggers, it is very common to see nested stored procedure calls. Automated migrations may not be able to handle error handling for nested stored procedure invocation. Therefore, it is necessary to pay close attention to error handling, especially for nested stored procedure invocations.

2.3.3 Using temporary tables extensively:

Some database developers use temporary tables extensively to simplify queries and avoid writing a complex query involving several tables. Early versions of some databases also had restrictions on the number of tables that could be joined in a query efficiently. Therefore, migrating stored procedures with lots of temporary tables warrants a closer look so that they can be avoided and can result in simplified code that leverages the native features of an Oracle database. Typically, migration tools maintain a one-to-one mapping of temporary tables during migration from one database to another. But important stored procedures which are executed very often and have demanding performance requirements should be examined thoroughly to eliminate unnecessary temporary tables in the new environment.

2.3.4 Converting stored procedures into functions:

The Oracle database does not support returning results to callers using the RETURN verb in stored procedures. This verb is only allowed in Oracle stored functions and not in stored procedures. However, it is very common to find Sybase and Microsoft SQL Server stored procedures using the OUT parameter as well as the RETURN verb to pass values and data to the caller. Converting these stored procedures into functions in Oracle also results in a different call signature (i.e., the syntax for executing a stored procedure versus executing a stored function is different because stored functions in Oracle must return a value).

2.3.5 Determining the impact of stored procedures returning result sets on Java applications (JDBC):

The Oracle database returns result sets to caller programs via explicitly defined OUT variables in stored procedures. However, other data- bases return multiple result sets implicitly, without having to declare variables to do so. This results in additional changes to Java programs when migrating to Oracle, such as declaring additional variables, binding, and explicit access of these variables for result set data.

2.4 Application Migration:

Application migration or porting can result from either migrate an application from one environment to another due to a complete rewrite, or simply from an underlying database platform that is being migrated to a new platform such as Oracle. Typically, application development falls into two categories:

2.4.1 Customized application development:

In this category, applications are generally developed in-house, by IT organizations, to support business functions. These applications almost always try to leverage all the native features of the database platform, as well as other IT systems in the organization, to drive maximum performance and tighter integration. As a result, applications tend to be heavily dependent on the database platform in which they were initially developed. As a result, any change to the database platform may result in changes to the applications. Features and functionalities leveraged by these applications also depend on the developer's skill set. Developers try to use the features they are most comfortable with. Once an application becomes obsolete due to a lack of the skills required to maintain its features, or due to the application becoming too brittle to add new features, the application is migrated to a new environment.

2.4.2 Generic application development (or packaged applications):

Typically, this category applies to independent software vendors (ISVs). ISVs develop generic application software that caters to a particular industry or a vertical market. They also tend to develop applications that do not depend heavily on the database. In fact, major ISVs offer versions of applications based on a particular database platform. Migration of a packaged application from one database to another involves installing and configuring the new version of the packaged application and importing the data and all the customizations from the original application. This is by no means a trivial task, because thorough testing needs to be done after the migration. From time to time, ISVs are forced to add support for new data- bases to their application software due to customer

demand. They are also under pressure to maintain a single or as few codebases as possible to reduce the effort involved in managing multiple codebases, each catering to a different database, because this means that if they have to implement a new feature, they will have to modify all the application codebases in a similar fashion and ensure consistency across them. From a migration perspective, customized applications are always migrated to new database platforms fully, because there is no need for them to support both the old and new database platforms in the long run. These applications can be changed to take full advantage of the new database platform. But ISVs need to support all existing database platforms, even as they add support for new databases. So, for them, it becomes a porting effort because they are simply adding more code to an existing application so that it will also work with the new database. ISVs try to reduce the application software codebase by using conditional coding practices such as conditional branches to a different piece of code, depending on the database platform on which it is deployed. Very large enterprise resource planning (ERP) software packages usually have separate codebases for each database.

As we mentioned when we were discussing the migration assessment phase, understanding the impact of database platform migration on applications is very important. Applications depend on the database platform in many ways:

2.4.3 Database-specific connection information:

Every database requires certain information to establish a connection with it. In the event of a database change, this information has to be updated in the applications that connect to a specific database. If every single program in an application connects to the database directly, instead of relying on a central database access layer, this otherwise trivial task becomes a challenge. This task can be automated through the use of scripts from the operating system to search and replace appropriate connection strings in application programs.

2.4.4 Use of database:

Specific parameters ODBC/JDBC drivers for database vendors have different parameters to support different requirements, such as transaction control, date/timestamp formats, and so forth. The Oracle JDBC driver, by default, enables AUTO COMMIT on a connection. This might create problems, especially when calling a database stored procedure in Oracle which leverages global temporary tables. Having set the AUTO COMMIT by default, the data in temporary tables will be deleted after any data manipulation statement (INSERT, DELETE, or UPDATE). To avoid this scenario, AUTO COMMIT for a JDBC connection should be explicitly disabled. For example:

Connection set Auto Commit (False)

- Using database-specific SQL statements with proprietary extensions requires changes when the database platform changes. It is a big challenge to identify how many application programs need to be changed because of their usage of SQL statements that do not conform to American National Standards Institute (ANSI) SQL standards or that are not supported by the Oracle database. In the assessment phase, there is a great deal of emphasis on identifying such programs and their database inter- actions in general (i.e., calling stored procedures, result set processing, embedded SQL usage, etc.).

- Invoking database stored procedures and functions that return result set Applications using ODBC/OLEDB drivers generally do not need to be modified when the database is migrated to Oracle. However, as of the latest release of Oracle Database 11g R2 (11.2.0.1), Java applications using the Oracle JDBC driver invoking stored procedures returning result sets from the database need to be modified to accommodate Oracle-specific requirements in terms of including bind variables for result sets, processing of multiple result sets, and similar functionality. Hopefully, these changes will not be necessary in future releases of the Oracle database.
- APIs for manipulation of large objects: There are differences in JDBC APIs used for manipulating large objects in Oracle as compared to databases such as Informix.

REFERENCES

[1]. F.A. Alvi1, B.S Choudary,N. Jaferry, E.Pathan.- A review on cloud computing security issues & challenges.

[2]. B.Meena,Krishnaveer ,Abhishek Challa- Cloud Computing Security Issues with Possible Solutions.

[3]. Dikaiakos, M.D; Katsaros, D.; Mehra, P.; Pallis, G.; Vakali, A.; (2010), "Cloud Computing Distributed Internet Computing for IT and Scientific Research".Vol.13 ,pp 10, Sept.-Oct. 2009.

[4]. Shuai Z; Shufen Z; Xuebin C; Xiuzhen H; (2010), "Cloud Computing Research and Development Trend", 2nd International conference on Future Networks, 2010. ICFN ' 10. pp 23, 22-24 Jan 2010.

[5]. Chang, L, Ti ; Chin L; Chang, A.Y.; Chun J, C;(2010), " Information security issue of enterprises adopting the application of cloud computing", IEEE 2010 Sixth International Conference on Networked Computing and Advanced Information Management (NCM),pp 645, 16-18 Aug. 2010.

[6]. R. Maggiani; (2009), "Cloud computing is changing how we communicate," 2009 IEEE International Professional Communication Conference, IPCC 2009,Waikiki, HI, United states ,pp 1, 19-22 July.

[7]. Geng L; David F; Jinzy Z; Glenn D; (2009), "Cloud computing: IT as Service, "IEEE computer society IT Professional", Vol. 11, pp.10-13, March-April 2009.

[8]. Basit Ali; (2009), "Ufone Launches Uconnect", published in TelecomPK.Net,12 August 2009.

[9]. Muzzammil Sheikh; (2011), "PTCL Launched EVO USB become Wi-Fi Hotspot", The Frontier Star (Northwest Frontier Province, Jan 26 2011 Issue.

[10]. Grobauer, B.; Walloschek, T.; Stocker,E.;(2011), "Understanding Cloud Computing Vulnerabilities",5487489 searchabstrSecurity & Privacy, IEEE, Vol 9, pp 50.

[11]. Gansen Z; Chunming R; Jin L; Feng Z; Yong T; (2010),,"Trusted Data Sharing over Untrusted Cloud Storage Providers",2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), pp 97, Nov. 30 2010-Dec. 3 2010.

[12]. Pearson, S.; (2009), "Taking account of privacy when designing cloud computing services",5071532 searchabstract CLOUD '09. ICSE Workshop on Software Engineering Challenges of Cloud Computing, 2009. pp 44, 23-23 May 2009.

[13]. Kresimir P; Zeljko H; (2010), "Cloud computing security issues and challenges", MIPRO 2010, May 24-28, 2010, Opatija, Croatia.

[14]. Minqi Z; Rong Z; Wei X; Weining Q; Aoying Z; (2010),"Security and Privacy in Cloud Computing: A Survey", Sixth international conference on Semantics Knowledge and Grid (SKG), pp 105, 1-3 Nov. 2010.

[15]. Popovic K; Hocenski Z; (2010), "Cloud computing security issues and challenge", 5533317searchabstractMIPRO, 2010 Proceedings of the 33rd International Convention , pp 344,24-28 May 2010.