Research Article

Data Migration From On Premise Oracle Database To SQL Manage Instance On Azure Cloud Using Azure Data Factory - A Working Approach

[1]Milind Jadhav , [2]Dr. Amol Goje, [3]Jitendra Chavan

1Department of Research
Neville Wadia Institute of Management Studies and Research
Pune,India
milindmay9@gmail.com
2Department of Research
Neville Wadia Institute of Management Studies and Research
Pune,India
acgoje@gmail.com
3Department of Information Technology
Marathwada Mitra Mandal College of Engineering
Pune,India
jitendra.rchavan@gmail.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

***Abstract*—** Data Migration has become important aspect nowadays when it comes to data movement from on premise databases to cloud storage or cloud databases. In this paper we present a working case study using cloud based ETL tool known as Azure Data Factory used for Data Migration from on premise Oracle database to cloud based SQL Managed Instance database for an organization. This paper evaluates the implementation of data migration process in general and specific to the tool and technologies involved in data migration process using Azure Data Factory for an organization. When an organization needs to move their application to cloud, the essential of data migration needs to be discussed, proper architecture is required to further break down each task to migrate the data. The proof of concept should be established to see if data is not getting truncated/altered in the process of migration and existing logic on the on premise database works well after moving data to the cloud. In this paper we also discuss about encryption process involved while migrating data as this is an important aspect in data migration to migrate data with existing algorithms used in on premise database and its implementation while data movement takes place using Azure Data Factory. In Oracle there are encryption algorithms being used to store sensitive user data, we have to analyze existing encryption/decryption process and implement an architecture with the help of data migration tool so that data remains intact after movement. Developer has to develop testing strategies to compare on premise data versus the data moved to the cloud storage. Azure Data Factory is powerful cloud-ETL tool to move your hundreds of table data at a time to new cloud database with maximum data transfer throughput. This data migration process requires thorough evaluation of multiple factors e.g. actual table size in migration, throughput, Virtual Machine used for data transfer, network bandwidth etc.

Keywords — Cloud, ETL, Data Migration, Azure Data Factory, SQL

I. INTRODUCTION

Consider a scenario where an organization wants to move their application to cloud and now they have come to the data aspect of movement. There would be mainly schema migration [table schema, indexes, constraints etc.], code migration [stored procedures, functions, triggers etc.] and finally data migration activity. Until and unless we have data in the target system, testing cannot be started. So data migration is an important aspect while moving your application to cloud environment. To speed up migration activity Microsoft has introduced Azure Data Factory [1] cloud –ETL [6] tool. ADF can connect to variety of source systems with more than 90 built in connectors. This paper can be used as a reference when developers, solutions architects, technical project managers need to evaluate ADF as a platform for data migration from On Premise DB to Microsoft Azure Cloud.

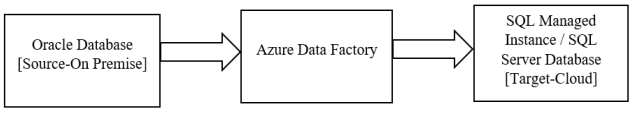

Fig.1 ETL process – On Premise to Cloud Migration

As shown in Fig.1 Source System is Oracle Database and Target System is Database on Cloud – SQL MI/ SQL Server DB [9]. The first phase in migration process is object migration where all the table schemas are migrated to target database. The DDL's are used for the creation of required application tables in target database. Once the

table object migration is completed then Stored Procedures, Triggers and Functions have also to be re-written to make them compatible with target database system. Again manual process will take time to convert each object manually, there is a tool known as SSMA or SQL Server Managed Assistance [9] which helps to convert these objects in bulk, package wise. These tools helps to convert the code 40-60%, depends on various programing constructs used inside the objects body. Here, the role of a database developer is very important to convert the remaining code into target database format.

II. THE ROLE OF AN ETL TOOL

ETL is a process of data extraction from source system, transforming the data with respect to business rules and finally loading it into the target system. Data is extracted from variety of sources. The main source of data extraction is RDBMS. Apart from RDBMS system data can be read from Flat Files, XML's, CSV's, JSON and many more depending applications requirements. While reading the data from source system, proper source definitions needs to be created to hold the metadata of sources in the ETL tools own data repository and start read process. Once data extraction is aligned, transformation or business rules are applied based on the data to be populated in the target system. After applying transformations data is loaded into the target system with different strategies e.g. Insert only, Update, Insert else update etc. Again the target operations depends on the on the type of target system.

An ETL tool is a software which is capable of performing these activities. These tools are capable of connecting source/target databases through connection parameters configured inside the tool. Data Migration to the cloud has become easy with the help of ETL tools. Application can be migrated to the cloud environment with minimum downtime and well established methods of data validation.

III. IMPORTANT COMPONENTS OF AZURE DATA FACTORY FOR DATA MIGRATION

A. Data Factory

Data Factory [4] is an user interface to manage all activities related to your data migration. Based on number of environments one can configure data factories. E.g. Dev, QA, Prod. Data Factory helps the developer to configure the entire ETL process for data migration. It connects to Source/Target database system through the Linked Services. Linked Services helps to connect to various source/target system involved in data migration. It also helps to configure secure connections to these system with the help of Azure Key Vault. The develop needs to pass configuration details like connections string etc. and save those details to establish connection to the source/target system. Data Factory also connects to Azure Blob Storage configuration to upload the files required for data migration.

B. Pipeline

The pipeline is the basic component used for data movement. ADF provides drag and drop facility to create a pipeline using various pipeline activities. ADF uses a user friendly interface to create the pipelines. A developer creates required pipelines in their own branch and then publishes it to the master branch. Since master branch will be used across all environments as baseline. Inside a pipeline developer can connect the activities e.g. Lookup Activity, Copy Activity and create a data movement pipeline. The logical order decides which component will execute first and perform data load process.

C. Activity

An activity is smallest component used inside the pipeline for data migration. Copy activity is most commonly used to read database from source and write it to the target database.

IV. CREATING AN ARCHITETURE FOR DATA MIGRATION

Creating a configuration table is an important step in creating architecture for data movement through ADF. When migrating number of tables , developer can create a configuration table with appropriate columns to identify the source , target , query etc. Using this configuration table developer can parameterize the source table name and target table name. This parameterization helps copy activity to pick the source and load into the target table dynamically. Here 'forEach' activity in ADF plays an important role. We can iterate through all the tables which are part of our configuration table and start loading one by one into target database. As per standard Microsoft documentation each pipeline can have maximum 50 tables in parallel for 'forEach' activity to handle. Suppose we have 500 tables so we can create 10 pipelines with similar logic. In fact we can create a pipeline which will be responsible for data load of 50 tables and pass those list of tables through configuration table. ADF has capability to reuse similar pipeline by 'invoke' feature. The configuration table helps achieve this by having a column to pass value for particular pipeline. E.g. if we have column in configuration table as 'pipeline number' with its value as 1 , it will load all tables with 'pipeline number = 1'. Hence we can execute pipelines from 1 to 10 in parallel and load all 500 tables.

V. DATA VALIDATION

Data Validation [3] [7] is an important activity in data migration process. It is vital that data load must be verified on execution of parallel runs on both existing system versus new system and compared against each other to

identify the differences occurred during migration process. Data validation should be done on row by row basis, incremental, full load capabilities as well. Certain tools help data validation easier. To test functionality or implementation logic of transformations SQL plays a major role when source and target are RDBMS [8] system. Queries have to be developed to validate data incrementally. Incremental logic will be based on number of records inserted in Source vs Target for the same Primary Keys on both sides on the same date criteria. Count of number of records should also match on both the sides. SQL queries have to be written to identify duplicate records, if there are duplicates in target system then proper analysis has to be made and required action to be taken. It might require a code fix to fix the issue post identification in data validation or it might be a data issue which can be ignored with proper comments. For instance it can be compared with production data. During the object migration process there are changes made to match with the source system data because of which data loss might occur. Wherever such changes have been made those columns need to be identified and tested accordingly and common guidelines can be prepared to speed up data validation for objects.

VI. TABLE CONSTRAINTS

As this paper mainly talks about cloud migration from legacy RDBMS to RDBMS on cloud, concept of column level constraint are available at both the side. The migration using Azure Data Factory requires disabling these constraints, otherwise ADF pipelines will fail with Primary Key / Foreign Key constraint violation issues. There is a way to handle this scenario. Before starting the load process constraints should be disabled. Apart from PK/FK constraint [10], there are indexes and triggers as well which needs to be disabled. Stored procedures needs to be written to disable constraints and through ADF Stored Procedure activity it can be called. If list of table objects are available then procedures are prepared for different type of constraints. For SQL MI database there is a generic procedure also made available to be used directly but proper care should be takes as it disables all the table constraint for all available tables under that database instance.

VII. SYSTEM ARCHITECTURE

A. Master Pipeline

Creation of a master pipeline requires to set up an essential configuration so that when master pipeline is triggered it starts the actual data movement for the tables which are part of the entire migration process. In the first phase all constraints are disabled. In the second phase of the master pipeline, master configuration table should be loaded. This table serves as the baseline for migrating entire table list. The parallel pipeline is triggered in the next phase, where master pipeline waits for the parallel pipeline to complete all the execute activities. Once parallel pipeline is completed all the constraints are enabled in the next phase which were disabled in the first phase. This summarizes architecture for the Master Pipeline.

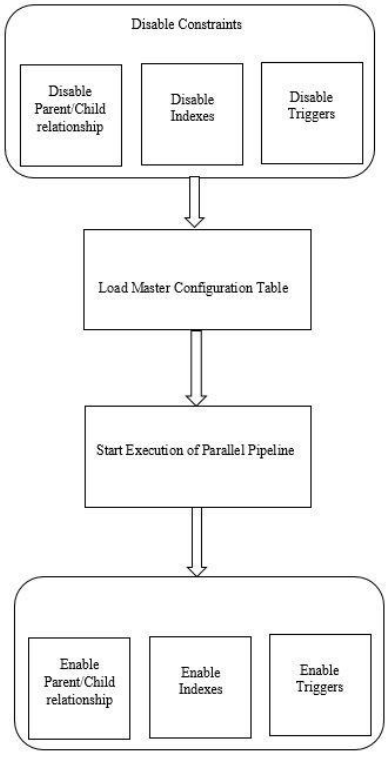

Fig. 2 Architecture Diagram

TABLE I. SAMPLE MASTER CONFIGURATION TABLE

B. Parallel Pipeline

In the third phase of master pipeline, parallel pipeline is triggered. This pipeline executes the number of execute activities configured for data migration process. Azure Data Factory provides this feature to start data migration for all the tables at a time. Once the largest running execute ectivity completes it will trigger the next activity in the master pipeline.

C. Base Pipeline

The parallel pipeline triggers the base pipeline through the Execute activity. Execute activity invokes the base pipeline consists of two important activities.

- Lookup Activity
- ForEach Activity

The lookup activity fetches list of tables to be loaded as part of particular batch. Batch is the ADF pipeline parameter for the base pipeline. For example, if Batch=1 in the Master_Configuration table then it will start loading all the tables in that particular batch. Using @item iterator variable of ForEach activity, it iterates over all the Source_Table_Names and load data into corresponding target tables specified in the Target_Table_Name column of the Master_Configuration table.

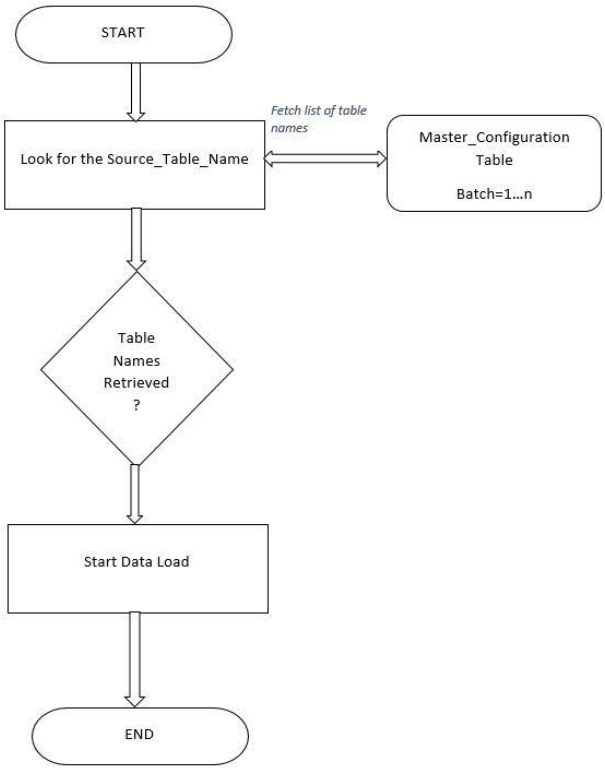

Fig. 3 Design of a base pipeline

By invoking base pipeline for all the batches code reusability is achieved.

Sample Configuration table	**Configuration Table**			
	Source Table Name	***Sink Table Name***	***Batch Number***	***Active Indicator***

VIII. CHALLENGES IN DATA MIGRATION

Migration of data from an existing system to the cloud database comes with many challenges using Azure Data Factory. The broad categories are listed below since explanation of these points are out of scope of this paper.

A. Calculating data migration timelines
B. CPU Utilization, Integrattion Runtime Requirements
C. Project Deadlines
D. Encryption challenges
E. Datasize and Production Environment Estimation

REFERENCES

1. https://docs.microsoft.com/en-us/azure/data-factory: "Mircosoft Official Website"
2. Erik Kajáti, Martin Miškuf, Peter Papcun : "Advanced Analysis of Manufacturing Data in Excel and its Add-Ins" SAMI 2017 • IEEE 15th International Symposium on Applied Machine Intelligence and Informatics • January 26-28, 2017 • Herl'any, Slovakia
3. Pooyan Jamshidi, Aakash Ahmad, and Claus Pahl: "Cloud Migration Research: A Systematic Review" IEEE TRANSACTIONS ON CLOUD COMPUTING, VOL. 1, NO. 2, JULY-DECEMBER 2013
4. Sudhir Rawat Abhishek Narain : "Understanding Azure Data Factory Operationalizing Big Data and Advanced Analytics Solutions" Apress
5. Claus Pahl, Huanhuan Xiong, and Ray Walshe : "A Comparison of On-Premise to Cloud Migration Approaches" IC4, Dublin City University Dublin 9, Ireland
6. Qin, Hanlin Jin, Xianzhen; Zhang, Xianrong : "Research on Extract, Transform and Load(ETL) in Land and Resources Star Schema Data Warehouse" 2012 Fifth International Symposium on Computational Intelligence and Design
7. Manel Souibguia, Faten Atiguib, Saloua Zammalia, Samira Cherfi, Sadok Ben Yahiaa : "Data quality in ETL process: A preliminary study" 23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems
8. Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo : "The Oracle Problem in Software Testing: A Survey" IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 41, NO. 5, MAY 2015
9. https://docs.microsoft.com/en-us/azure/azure-sql/managed-instance/sql-managed-instance-paas-overview : "Mircosoft Official Website"
10. Lijun Ma : " The integrity rules and constraints of database" Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology