Data Synchronization Across Disparate Database Management Systems

Chirag Goel[1]
[1]Sr. Database Administrator, Illinois Institute of Technology, Chicago, USA

Abstract- Database management systems are packaged software which defines, manipulate, retrieve and manage data in a database. In the market we have a good number of DBMS supported by different vendors and number keeps on increasing every year. Very few provide ways to synchronize data between disparate DBMS, and that is sometimes limited to only importing from external DBMS. Vendors do this due to various reasons like they don't have control over the code of other DBMS etc. Here we presented an approach and design along with implementation in one of the DBMS. The approach defined in this article can be applied to any DBMS.The design here will enable users to synchronize data between one or more tables from one or more DBMS systems. This will also enable the user to implement near-real-time change data capture which can keep tables sync in the same DBMS or disparate DBMS.
Keywords- Database Management System, replication, Data Sync, Distribution, Change Data Capture.

I. INTRODUCTION

Database management systems provide the ability to define, manipulate, retrieve and manage data in databases. With the increasing demand of functionalities and flooding of data, DBMS are becoming more intelligent with the time. At present we have various DBMS which are getting used at high scale in enterprises. Most used features are present in all of them in one form or other and with the same or different names. Business needs for porting data from one database system to another has been well served by ETL tools and other change data capture techniques available. Even after so much advancement we still don't have proper ways to keep data across disparate DBMS in synchronization. Some DBMS does have the feature to replicate data from other limited sources with different limitations applied to it but still, there is a need to have some algorithm that can be applied to any DBMS without any limitation.

II. PROBLEM STATEMENT

Disparate DBMS in enterprises are used to manage disparate data sources. Consider an example of the finance industry, this industry is getting a different kind of data from different sources, which can be credit bureau, credit card type, credit card holders, checking accounts, and so on. With time this organization owned different platforms like mainframes, AIX, Windows,etc and each of them holds different DBMS such as DB2, SQL Server, Oracle, MySQL, etc. ETL tools are used to migrate data from one database table to another database table which can truncate and load. The problem statement for which this research is providing a solution is how one can perform real-time or near real-time change data capture among disparate database platform tables.

III. PROPOSED SOLUTION

This research proposes a solution for keeping data synchronized between the same database system and disparate database systems. The proposed solution provides the capability for near real-time synchronization among one or many tables. The approach has been presented with real-time implementation on SQL Server along with SQL server integration services as an ETL tool. Implementation has been regressively tested withsmall and large datasets. Scripts have been provided to facilitate readers in implementation and make the experience better. Performance has been monitored on the implemented system and measures have been taken to test the proposed solution for the real-world usage.

IV. ACTORS INVOLVED IN DESIGN

Below are the actors who are part of the design and will be needed to implement the approach.

1. Source Database
2. Source Table
3. Source Audit Table
4. Distribution Database
5. Distribution table
6. Target Database
7. Target Table

8. Target Audit Table
9. ETL tool
10. Scheduling tool

We are working with Microsoft SQL Server and using the ETL and scheduling tool which comes with the SQL Server suite. ETL tool used in the presentation is Visual Studio Data Tools and the Scheduling tool used is SQL Server Agent. Any DBMS, along with any ETL and scheduling tool can be used to implement this design. The design presented here is not limited by any DBMS, ETL, scheduling tool or any table size.

V. SOURCE DATABASE AND TABLES

The source database used in the demonstration is Admin, you can use any name. The table on the source database which will be replicated is tblOrders. tblOrdersAudit is created in the source database to keep track of any data inserted, updated and deleted.

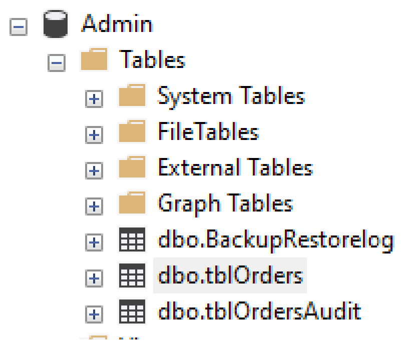


Figure 1. Source Database and Tables

VI. SCRIPTS FOR SOURCE DATABASE

Below scripts are used for setting up the source database. In general case source database will be existing as that will be part of the requirement from where data needs to be replicated. For our demonstration use below TSQL script to create tblOrders

```
create table tblOrders
(
OrderID integer Identity(1,1) primary key,
OrderApprovalDateTime datetime,
OrderStatusvarchar(20)
)
```

Figure 2. Create tblOrders

Below script creates tblOrdersAudit which will track any insert, update and delete on the source table.

```
create table tblOrdersAudit
(
OrderAuditID integer Identity(1,1) primary key,
OrderID integer,
OrderApprovalDateTime datetime,
OrderStatusvarchar(20),
performedActionnvarchar(50),
UpdatedBynvarchar(128),
UpdatedOn datetime
)
go
```

Figure3. Create tblOrdersAudit

Create below triggers on the source table. Three triggers are created tblTriggerAuditRecordInsert, tblTriggerAuditRecordUpdate, tblTriggerAuditRecordDel for insert, update and delete respectively.

```
create trigger tblTriggerAuditRecordInsert on tblOrders
after  insert
as
begin
insert intotblOrdersAudit
(OrderID,OrderApprovalDateTime,OrderStatus,performedAction,UpdatedBy,UpdatedOn)
selecti.OrderID,i.OrderApprovalDateTime,i.OrderStatus,'Inserted', SUSER_SNAME(),getdate()
fromtblOrders t
inner join inserted i on t.OrderID=i.OrderID
end
go
create trigger tblTriggerAuditRecordUpdate on tblOrders
after update
as
begin
insert intotblOrdersAudit
(OrderID,OrderApprovalDateTime,OrderStatus,performedAction,UpdatedBy,UpdatedOn)
selecti.OrderID,i.OrderApprovalDateTime,i.OrderStatus,'Updated',SUSER_SNAME(),getdate()
fromtblOrders t
inner join inserted i on t.OrderID=i.OrderID
end
go
create trigger tblTriggerAuditRecordDel on tblOrders
after delete
as
begin
insert intotblOrdersAudit
(OrderID,OrderApprovalDateTime,OrderStatus,performedAction,UpdatedBy,UpdatedOn)
selecti.OrderID,i.OrderApprovalDateTime,i.OrderStatus,'Deleted', SUSER_SNAME(),getdate()
fromtblOrders t
right join deleted i on t.OrderID=i.OrderID
end
go
```

Figure 4. Source Triggers

VII. DESTINATION DATABASE AND TABLES

Destination database created id replTarget and tables created are tblOrders and tblOrdersAudit. tblOrders will be a final table in which users will query as that is synchronized from the source. tblOrdersAudit is created to keep track and synchronize data in tblOrders.

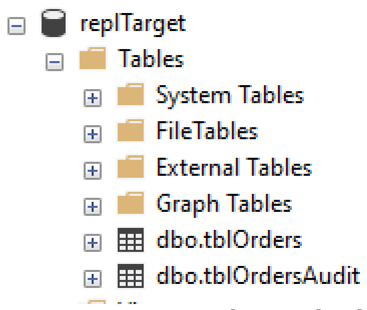


Figure 5. Target Database and Tables

VIII. SCRIPTS FOR DESTINATION DATABASE

Create a destination database name replTarget and staging table on Target. These tables will be scripted as per your requirement. If the source table is of a different name than destination tblOrders will be of the same name as the source table. To create an Audit table in destination take a script of the original table in question and add four new columns to the script. Four columns that need to be added in the Audit table are OrderAuditId, performedAction, UpdatedBy, and UpdatedOn.

```
Create database replTarget
Go
```

Figure 6. Target Database Script

```
create table tblOrdersAudit
(
OrderAuditID integer,
OrderID integer,
OrderApprovalDateTime datetime,
OrderStatusvarchar(20),
performedActionnvarchar(50),
UpdatedBynvarchar(128),
UpdatedOn datetime
)
Go
```

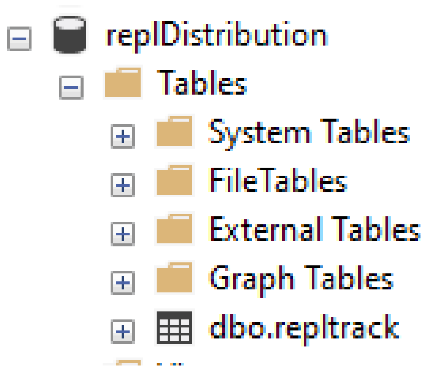
Figure 7. Destination tblOrdersAudit script

```
create table tblOrders
(
OrderID integer,
OrderApprovalDateTime datetime,
OrderStatusvarchar(20)
)
Go
```

Figure 8. Destination tblOrders Script

IX. DISTRIBUTION DATABASE

This database keeps track of AuditId. I named it replDistribution as I felt distribution matches with the purpose of this database.replTrack table will keep track of the table name and audit id for the respective table. Audit id is the id of the last row read from sources. This will get updated as more rows getto read from the source. Another important feature in this design is that this can accommodate any number of tables for the synchronization process. You will need to add a new entry to this table whenever you need to implement a new table and will need to change the rest of the scripts. Below TSQL script can be used to create this table.



```
replDistribution
  Tables
    System Tables
    FileTables
    External Tables
    Graph Tables
    dbo.repltrack
```

Figure 9. Distributor Database

X. SCRIPTS FOR DISTRIBUTION DATABASE

Repltrack in distribution is for tracking the last read id from the source. Here we have a table name column along with auditIdTracked which allows us to replicate as many tables as we want. This research is implemented with one table but many tables can be added here.

```
create table repltrack
(
replID integer Identity(1,1) primary key,
Tablenamenvarchar(250),
auditIdTrackedint
)
Go
```

Figure 10. Distribution repltrack Script

XI. ETL PACKAGE DESCRIPTION

The ETL tool I used here is SQL Server Integration Services. If a user hasa different ETL tool then you can implement the below steps and make sure they are in order. The package helps to keep data in sync between source and target. It reads Audit id from replTrack table and keeps that in the variable to be used through the package in different steps. Below I will describe each step in the package.

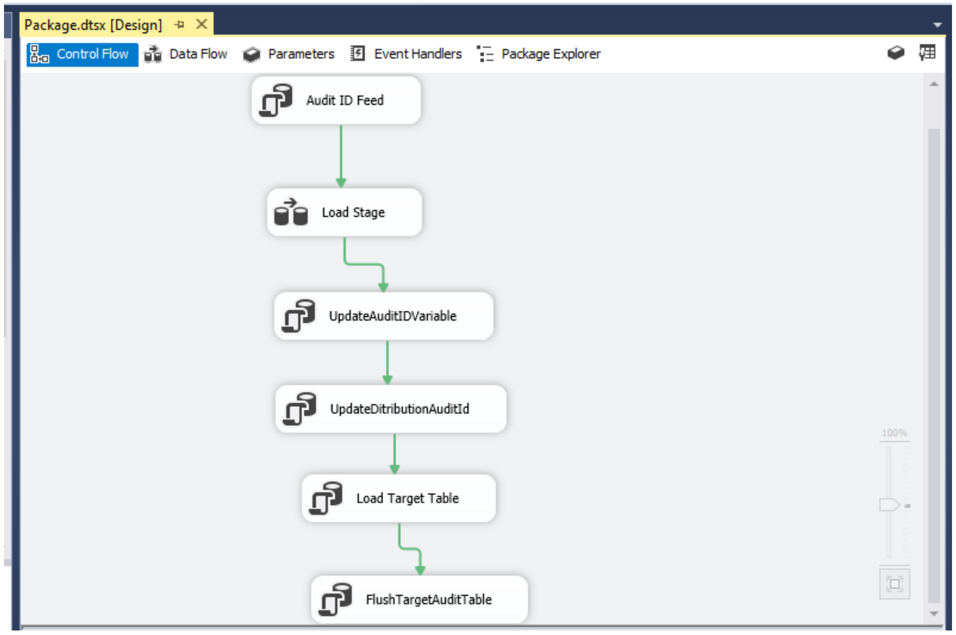


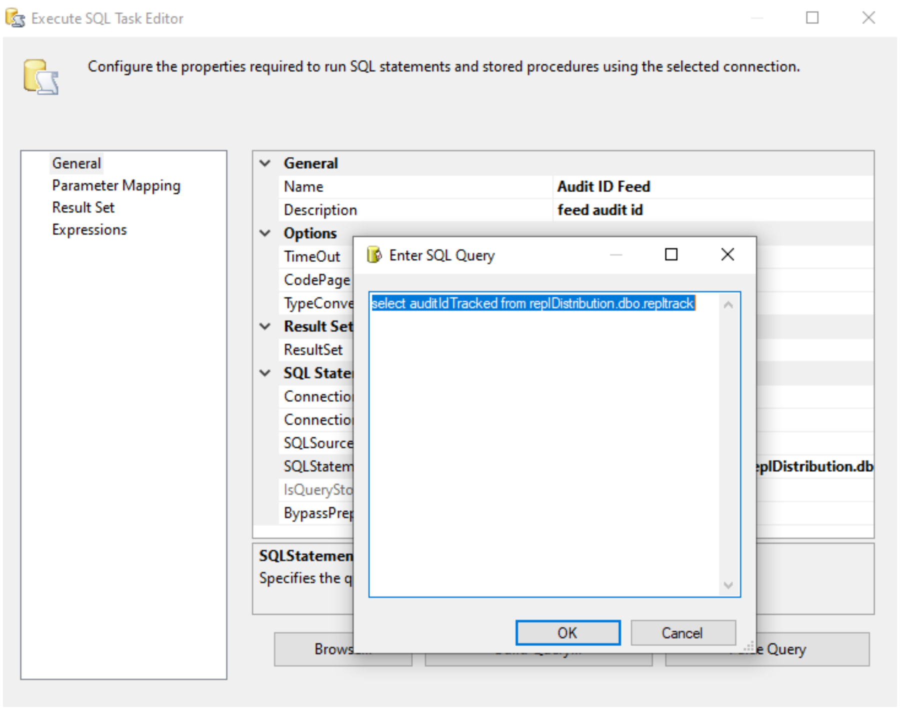
Figure 11. ETL Package Layout

11.1. Audit ID Feed–

In the first step, we get Audit feed id from the distributor database.

```
selectauditIdTrackedfromreplDistribution.dbo.repltrack
```

Figure 12. AuditId Feed Script


Execute SQL Task Editor

Configure the properties required to run SQL statements and stored procedures using the selected connection.

General
Parameter Mapping
Result Set
Expressions

General
Name: Audit ID Feed
Description: feed audit id

Enter SQL Query

```
select auditIdTracked from replDistribution.dbo.repltrack
```

Figure 13. AuditId Feed Task

11.2. Load Stage–

Load stage data by reading Source tblOrderAudit.We are passingauditIdTracked as a parameter so that we can get anything greater than audit id.

```
1.   select*fromAdmin.dbo.tblOrdersAudit
2.   whereOrderAuditID>?
3.   order byOrderID,OrderAuditID
```

Figure 14. Source Script for Load Stage Task

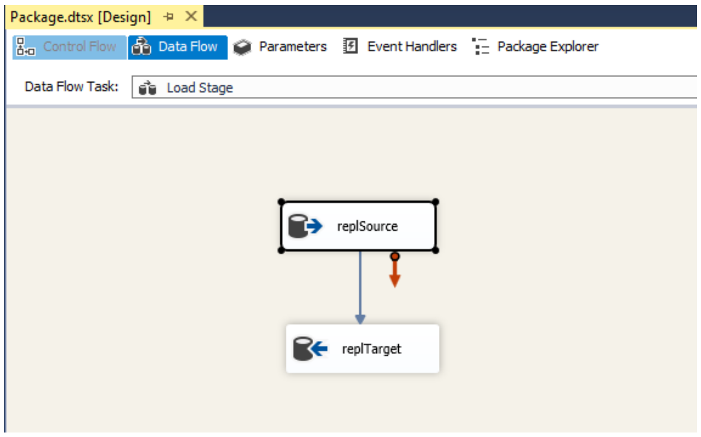
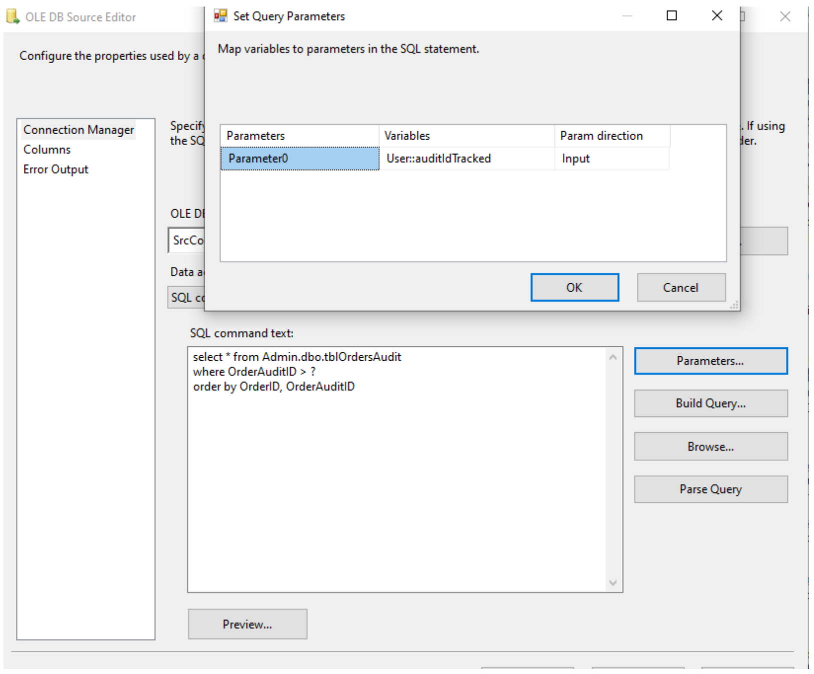


Figure 15. Load Stage Task


OLE DB Source Editor

Set Query Parameters

Map variables to parameters in the SQL statement.

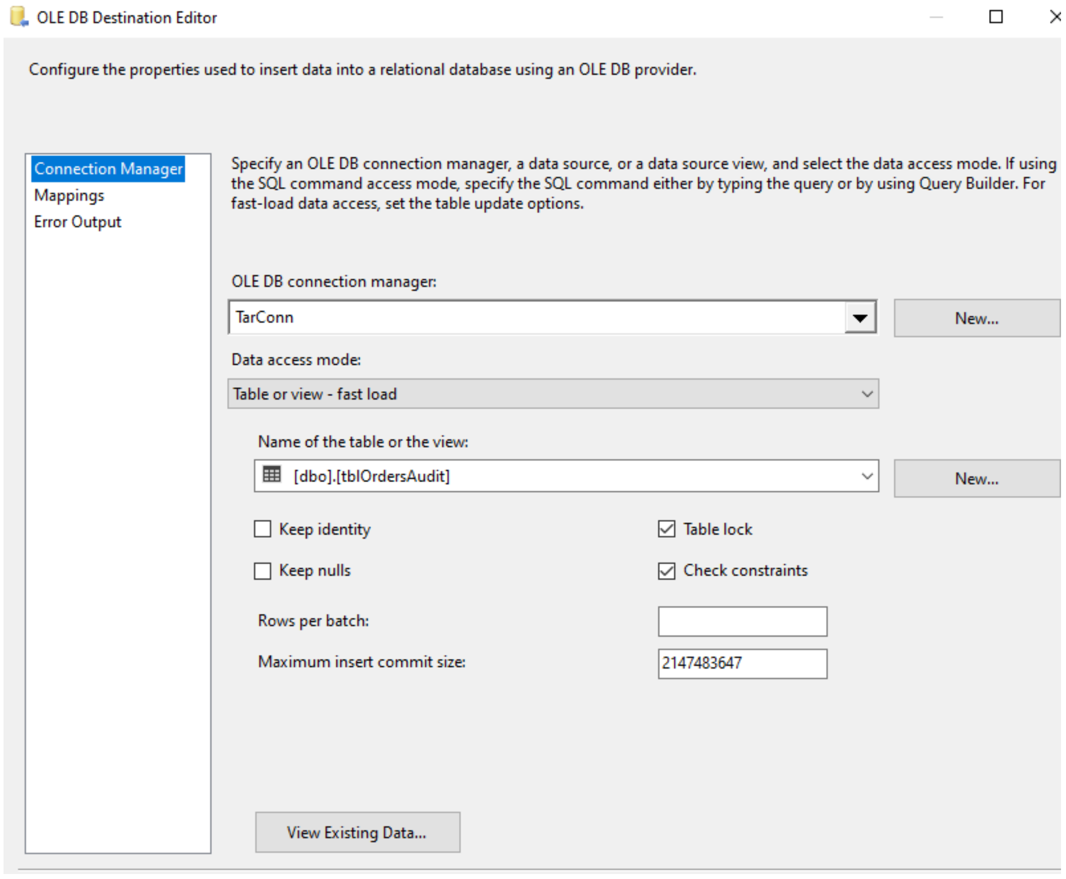
Parameters	Variables	Param direction
Parameter0	User::auditIdTracked	Input

SQL command text:

```
select * from Admin.dbo.tblOrdersAudit
where OrderAuditID > ?
order by OrderID, OrderAuditID
```

Figure 16. Parameter in Load Stage Task

In Destination we are inserting data in tblOrdersAudit


OLE DB Destination Editor

Configure the properties used to insert data into a relational database using an OLE DB provider.

Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder. For fast-load data access, set the table update options.

OLE DB connection manager: TarConn

Data access mode: Table or view - fast load

Name of the table or the view: [dbo].[tblOrdersAudit]

- [ ] Keep identity
- [ ] Keep nulls
- [x] Table lock
- [x] Check constraints

Rows per batch:

Maximum insert commit size: 2147483647

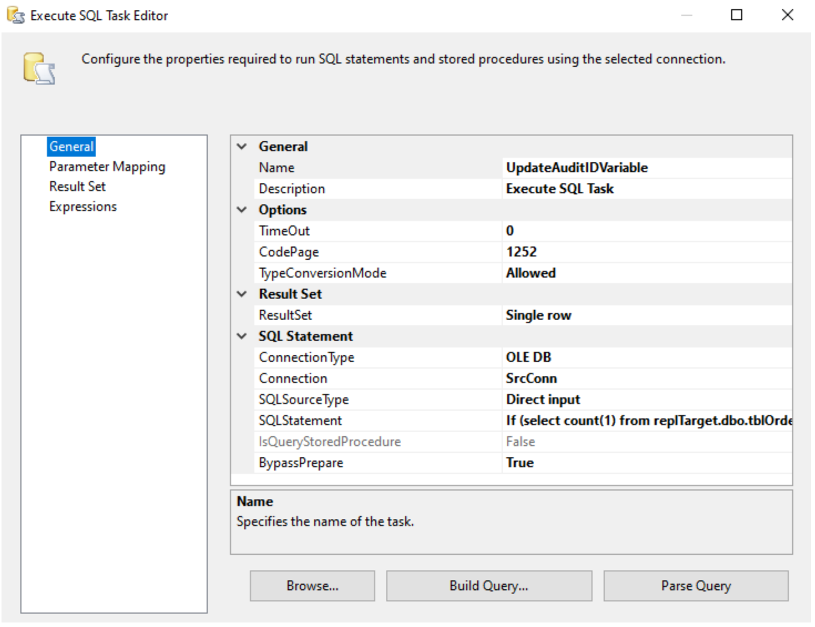
Figure 17. Destination in Load Stage task

11.3. Update AuditId Variable–

This step updatesAuditIdVariable. This checks for the count of rows in tblOrderAudit in the destination database and if the count is greater than 0 than it gets the maximum audited value and sets it for a variable used in package.

```
If(selectcount(1)fromreplTarget.dbo.tblOrdersAudit)>0
Begin
select TOP 1OrderAuditIDfromreplTarget.dbo.tblOrdersAudit
order byOrderAuditID desc
end
else
Select?as'OrderAuditID'
```

Figure 18. Update AuditId Script



Configure the properties required to run SQL statements and stored procedures using the selected connection.

General	
Name	UpdateAuditIDVariable
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	Single row
SQL Statement	
ConnectionType	OLE DB
Connection	SrcConn
SQLSourceType	Direct input
SQLStatement	If (select count(1) from replTarget.dbo.tblOrde
IsQueryStoredProcedure	False
BypassPrepare	True

Name
Specifies the name of the task.

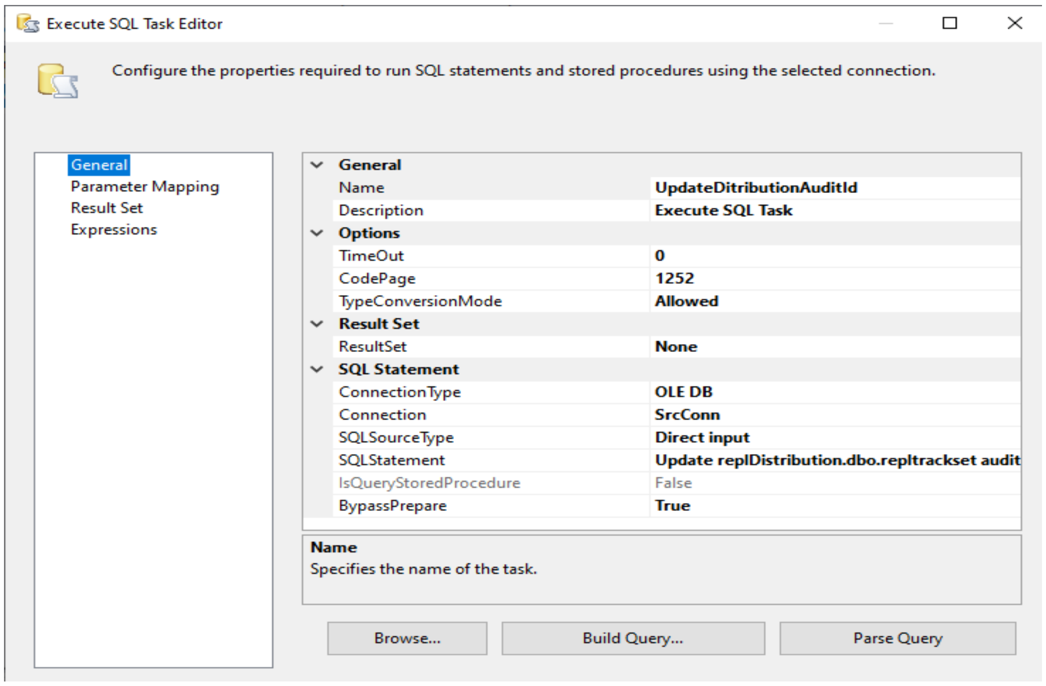
Figure 19. UpdateAuditIdVariable Task

11.4. Update Distribution Audit Id–

This step updates the replTrack table with audit id from the variable.

```
UpdatereplDistribution.dbo.repltrack
setauditIdTracked=?
```

Figure 20. Update distribution AuditId Script



Configure the properties required to run SQL statements and stored procedures using the selected connection.

General
Parameter Mapping
Result Set
Expressions

General	
Name	UpdateDitributionAuditId
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	OLE DB
Connection	SrcConn
SQLSourceType	Direct input
SQLStatement	Update replDistribution.dbo.repltrackset audit
IsQueryStoredProcedure	False
BypassPrepare	True

Name
Specifies the name of the task.

Figure 21. Update Distribution AuditId

11.5. Load Target Table–

The destination table gets loaded in this step. To load destination,the procedure has been created which read stage data from tblOrderAudit and feeds the tblOrder.

```
Execute replLoadTargetTable
```

Figure 22. Package Load Target Table Script

Below is the code for the procedure which is getting called in this step.

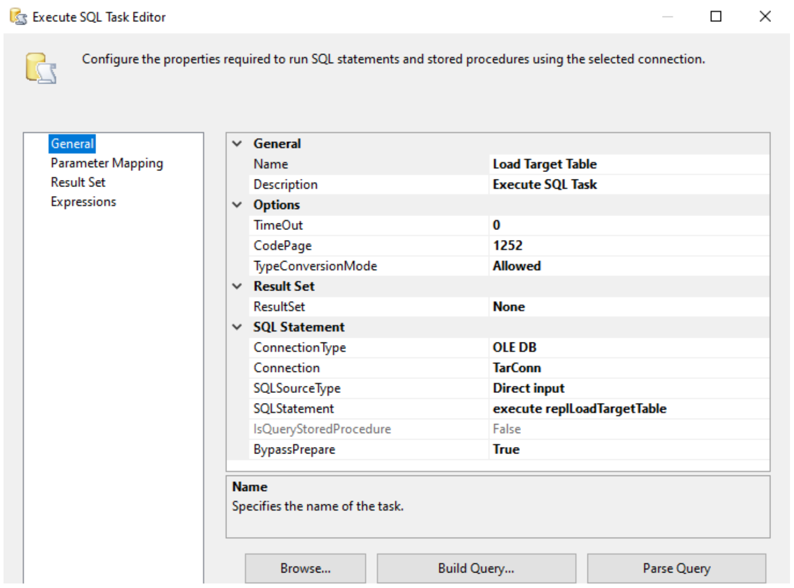
```
Create procedure [dbo].[replLoadTargetTable]
as
Begin

deletefromreplTarget.dbo.tblOrders
whereOrderIDin
(select75rderedfromreplTarget.dbo.tblOrdersAudit
          whereperformedAction=‘Deleted’)

insert intoreplTarget.dbo.tblOrders
selectOrderID,OrderApprovalDateTime,OrderStatusfromreplTarget.dbo.tblOrdersAudit
whereperformedAction=‘Inserted’

Update t
Sett.OrderID=ta.OrderID,
t.OrderApprovalDateTime=ta.OrderApprovalDateTime,
t.OrderStatus=ta.OrderStatus
fromreplTarget.dbo.tblOrders t
inner joinreplTarget.dbo.tblOrdersAudit ta
   on t.OrderId=ta.orderid
andta.performedAction=‘Updated’
End
GO
```

Figure 23. replLoadTargetTable procedure Script



Execute SQL Task Editor

Configure the properties required to run SQL statements and stored procedures using the selected connection.

General	
Name	Load Target Table
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	OLE DB
Connection	TarConn
SQLSourceType	Direct input
SQLStatement	execute replLoadTargetTable
IsQueryStoredProcedure	False
BypassPrepare	True

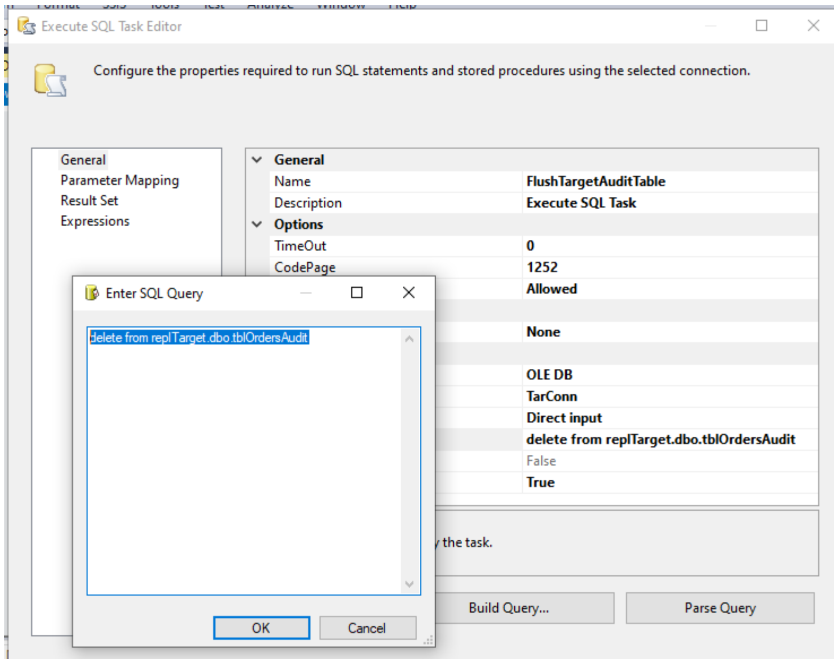
Name
Specifies the name of the task.

Figure 24. Load Target Table Task

11.6. FlushTargetAuditTable–
tblOrdersAudit table in destination is purely stage table which we truncate after loading target table in destination database.

```
deletefromreplTarget.dbo.tblOrdersAudit
```

Figure 25. FlushTargetAuditTable Script



Execute SQL Task Editor

Configure the properties required to run SQL statements and stored procedures using the selected connection.

General	
Name	FlushTargetAuditTable
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
	Allowed
	None
	OLE DB
	TarConn
	Direct input
	delete from replTarget.dbo.tblOrdersAudit
	False
	True

Enter SQL Query

```
delete from replTarget.dbo.tblOrdersAudit
```

Figure 26. FlushTargetAuditTable Task

XII. PACKAGE EXECUTION

Let’s see the execution of a complete set in action. Before this demonstration, I scheduled this package using the SQL Server agent and it is getting executed every second. We can use any scheduler to run the packages and for the package itself, I am using SSIS as it comes packages with SQL Server but again any ETL tool can be used. Its totally on the discretion of users what they have available at their dispense. The frequency of executing this package is totally on the requirement and should be consulted with business to get SLA. In this demonstration, we will add data in the source and will see it getting applied to the destination.

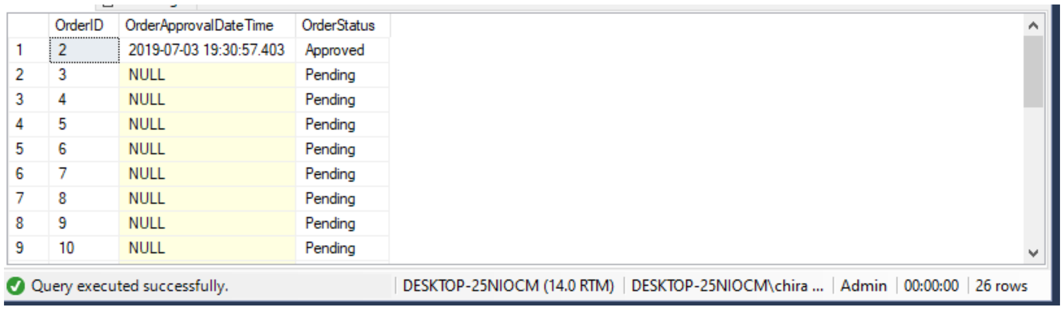
12.1 Add Row to the source table–

We already have 23 rows in the source table and adding below 3 will take the count to 26 rows.

```
insert intotblOrders values (NULL,'Pending')
insert intotblOrders values (NULL,'Pending')
insert intotblOrders values (NULL,'Pending')
go
```

Figure 27. Feed Source Table Script

3 rows got added in Source table

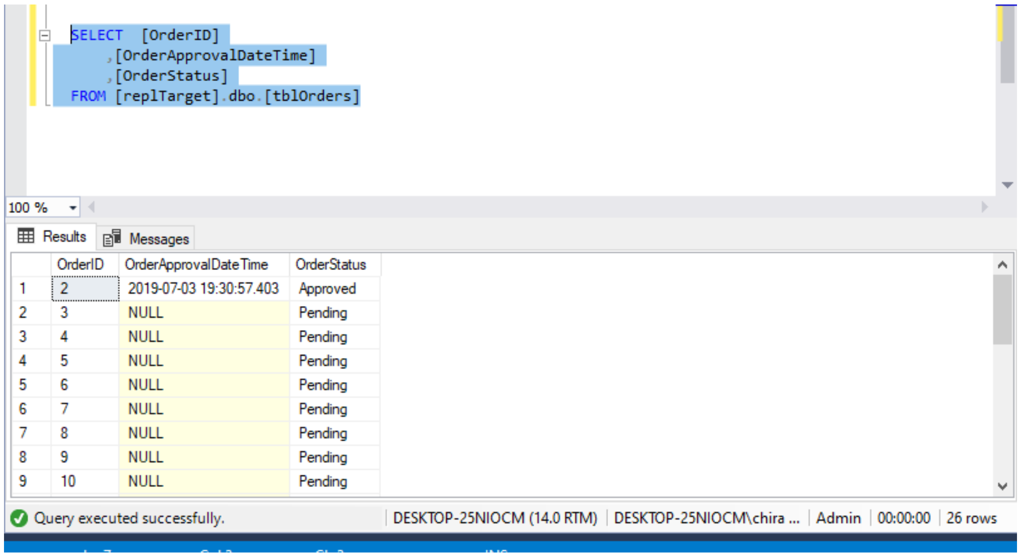


	OrderID	OrderApprovalDateTime	OrderStatus
1	2	2019-07-03 19:30:57.403	Approved
2	3	NULL	Pending
3	4	NULL	Pending
4	5	NULL	Pending
5	6	NULL	Pending
6	7	NULL	Pending
7	8	NULL	Pending
8	9	NULL	Pending
9	10	NULL	Pending

Query executed successfully. | DESKTOP-25NIOCM (14.0 RTM) | DESKTOP-25NIOCM\chira ... | Admin | 00:00:00 | 26 rows

Figure 28. Source Table after rows addition

Below presents results in the destination table after the insert has been performed on the source table. In the below right corner of the below screen, it will show 26 rows in the destination table which is the same as the source table.



```
SELECT  [OrderID]
      ,[OrderApprovalDateTime]
      ,[OrderStatus]
  FROM [replTarget].dbo.[tblOrders]
```

	OrderID	OrderApprovalDateTime	OrderStatus
1	2	2019-07-03 19:30:57.403	Approved
2	3	NULL	Pending
3	4	NULL	Pending
4	5	NULL	Pending
5	6	NULL	Pending
6	7	NULL	Pending
7	8	NULL	Pending
8	9	NULL	Pending
9	10	NULL	Pending

Query executed successfully. | DESKTOP-25NIOCM (14.0 RTM) | DESKTOP-25NIOCM\chira ... | Admin | 00:00:00 | 26 rows

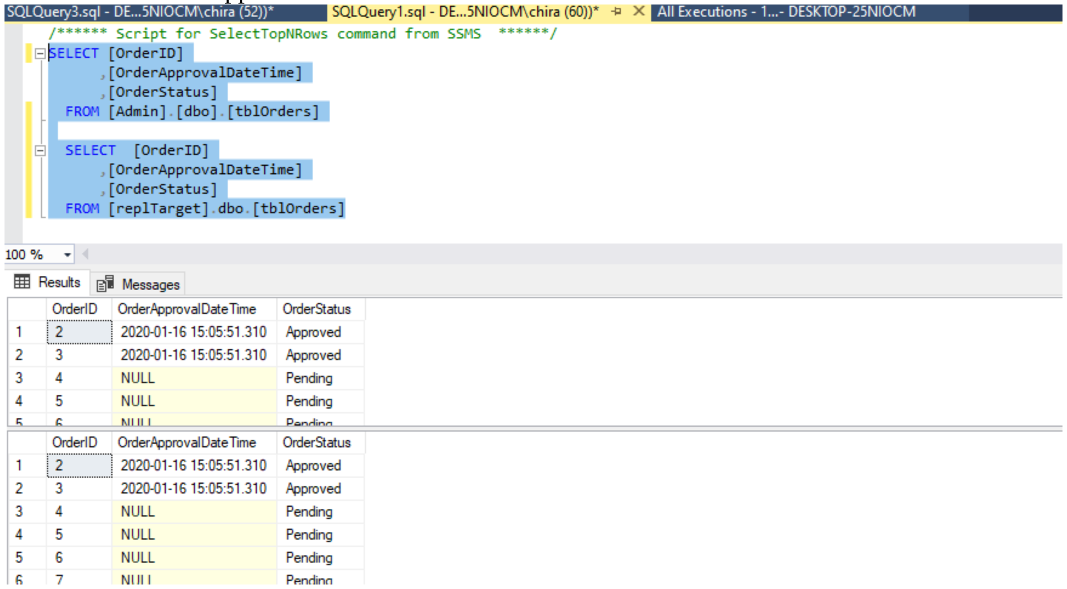
Figure 29. Destination Table After rows addition in Source table

Now let's update the status of order Id 2 and 3 to Approved–

```
update tblOrders
setOrderStatus='Approved',
OrderApprovalDateTime=getdate()
whereOrderIDin(2,3)
go
```

Figure 30. Update Source Table Script

The top section of the below screen contains a result for the source and the bottom contains the destination result set. Below show the same order approved date and order status for order id 2 and 3.



SQLQuery3.sql - DE...5NIOCM\chira (52))* | SQLQuery1.sql - DE...5NIOCM\chira (60))* | All Executions - 1...- DESKTOP-25NIOCM

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT [OrderID]
      ,[OrderApprovalDateTime]
      ,[OrderStatus]
  FROM [Admin].[dbo].[tblOrders]

  SELECT  [OrderID]
      ,[OrderApprovalDateTime]
      ,[OrderStatus]
  FROM [replTarget].dbo.[tblOrders]
```

	OrderID	OrderApprovalDateTime	OrderStatus
1	2	2020-01-16 15:05:51.310	Approved
2	3	2020-01-16 15:05:51.310	Approved
3	4	NULL	Pending
4	5	NULL	Pending
5	6	NULL	Pending

	OrderID	OrderApprovalDateTime	OrderStatus
1	2	2020-01-16 15:05:51.310	Approved
2	3	2020-01-16 15:05:51.310	Approved
3	4	NULL	Pending
4	5	NULL	Pending
5	6	NULL	Pending
6	7	NULL	Pending

Figure 31. Source and Target Table results after the update

In the above testing, we added new rows and updated rows in the source table and we noticed the destination table gets updated right away. We scheduled package to run after every second which allows it to check for rows in source table every second keep source and destination in sync. We also did regress testing with heavy data load as millions of rows which can be the use case when you are trying to sync tables in the datawarehouse. This proposed solution worked excellently with any amount of data and kept the destination table in sync.

XIII. CONCLUSIONS

Data synchronization between disparate systems is becoming very crucial with the fact that most organizations have various database management systems and need often arise where we need to sync data in various tables.
Native data sync options are being provided by vendors only for their DBMS and sometimes they extend it to different vendor DBMS but not all DBMS are supported.The proposed solution can be implemented on any DBMS.
Scripts provided in this article are for SQL Server only but those can be modified depending upon the DBMS in use.
The approach provided can be used for near real-time sync and also for scheduled sync of data between database tables on disparate DBMS.
SLA requirements should be check before implementing the proposed solution and SLA can be met by tweaking the ETL job schedule.

XIV. REFERENCES

[1] Jim Gray, “The Transaction Concept: Virtues and Limitations”, Proceedings of Seventh International Conference on Very Large Databases, June 1981.

[2] Cuzzocrea, A., Song, I.Y., Davis, K.C., “Analytics over Large-Scale Multidimensional Data: The Big Data Revolution!”, Proc. of the ACM 14th international workshop on Data Warehousing and OLAP (DOLAP '11), ACM, New York, USA, pp.101-104, 2011.

[3] Peng, Yi, Kou, Gang, Shi, Young, Zhengxin, Chen , “A descriptive framework for the field of Data Mining and Knowledge Discovery”, International Journal of Information Technology & Decision Making, Vol. 7, No. 4, 2008

[4] C. Goel, “Information Security Least Privilege Requirement Analysis for SQL Database Backups”, International Journal of Computer Trends & Technology - IJCTT , January 2020, ISSN: 2231-2803.

[5] Reddy, G. Satyanarayana, Srinivasu, Rallabandi, Rao, M., “Data Warehousing, Data Mining, OLAP and OLTP Technologies are essential elements to support decision-making process in industries”, International Journal on Computer Science and Engineering, 2(9), 2010

[6] Li, C., Wang, S. “A Data Model for Supporting On-Line Analytical Processing”, Proc. of the 5th International Conference on Information and Knowledge Management, pp. 81-88, 1996.

[7] Lujan-Mora, S., Trujillo, J., Song, I.-Y. , “A UML a profile for multidimensional modeling in data warehouses”. Data Knowl. Eng., 59(3), pp.725–769, 2006.

[8] Ordonez, C., Gracia-Alvarado, C., “A Data Mining System Based on SQL Queries and UDFs for Relational Databases”, CIKM’11, Glasgow, Scotland, UK,2011.

[9] Zaniolo, C., “Intelligent Databases: Old Challenges and New Opportunities”, Journal of Intelligent Information Systems, 1, pp.271-292, 1992.

[10] Cattell, R., “Scalable SQL and NoSQL Data Stores”, ACM SIGMOD Record, 39(4), pp. 12-27, 2010.

[11] Tabet K, Mokadem R, Laouar MR, Eom S.,“Data Replication in Cloud Systems: A Survey”. International Journal of Information Systems and Social Change. 2017 July-September; 8(3).

[12] Whitman ME, Mattord HJ, Green A. “Principles of Incident Response and Disaster Recovery”. 2nd ed.: Cengage Learning; 2013.

[13] Alhazmi OH, Malaiya Y. “Evaluating Disaster Recovery Plans Using the Cloud”. In Reliability and Maintainability Symposium (RAMS); 2013; Beijing: IEEE. p. 1-6.

[14] Connolly T, Begg C.,“Database system a practical approach to design, implementation, and management”. 5th ed. Boston: Addison-Wesley; 2009.

[15] “Database Language SQL-Part 2: Foundation (SQL/Foundation)”, ANSI/ISO/IEC International Standard (IS), 1999.

[16] Chan, M.Y. and Cheung, S.C. “Applying white box testing to database applications”. CSTR, Hong Kong University of Science and Technology, HKUST-CS99-01. 1999.

[17] A. B. Silberschatz, H. F. Korth, and S. Sudarshan., “Database System Concepts”. McGraw-Hill, sixth edition, 2011.

[18] A.P.G. Brown, "Modelling a Real-World System and Designing a Schema to Represent It", in Douque and Nijssen (eds.), Data Base Description , North-Holland, 1975, ISBN 0-7204-2833-5.

[19] J. D. Ullman and J. Widom. “A First Course in Database Systems”. Prentice Hall, third edition, 2008.

[20] Codd, E.F. (June 1970). "A Relational Model of Data for Large Shared Data Banks" .Communications of the ACM . 13 (6): 377–387.doi : 10.1145/362384.362685.

[21] C.J. Date. “An Introduction to Database Systems”. Addison Wesley, 1999.

[22] Shabani, I., Cico, B., Chaudhary, A. “Solving Problems in Software Applications through Data Synchronization in Case of Absence of the Network”.International Journal of Computer Science Issues, 2014.

[23] Kaur, A., Aashima. “Synchronized Algorithm For Database And Image Processing Between Client And Server”. International Journal of Computer Science and Information Technologies, 2014.

[24] S. V. Krishna, and S.Kumar M “Data Synchronization Using Cloud Storage” International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 11, November 2012.

[25] Hua Gu , Lei Huang ,Jing Liu Qiuli Qin “Research on the Data Synchronization of Cloud Stroke based on JSON” International Journal of Grid and Distributed Computing, Vol. 9, No. 7,2016, pp.201-210.