St. Cloud State University
theRepository at St. Cloud State

Culminating Projects in Mechanical and Manufacturing Engineering

Department of Mechanical and Manufacturing Engineering

8-2016

Integrating Online Wire Transfer Fraud Data with Suspicious Wire Transfer Data Using SSIS

Akhila Arramreddy
St. Cloud State University

Integrating Online Wire Transfer Fraud Data with Suspicious Wire Transfer Data Using SSIS

by

Akhila Arramreddy

A Starred Paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

in Engineering Management

August, 2016

Starred Paper Committee:
Hiral Shah, Chairperson
Ben Baliga
Balasubramanian Kasi

Abstract

The client is one of the banking systems which is in the process of providing online wire transfers to the customers. This project is about how the suspicious wire transfer data and confirmed wire transfer fraud data are handled. Currently both data are entered through two different applications and are stored separately. Since it is important for any business to make analysis how their business is running to make further decisions the data from these two applications should be integrated. This can be achieved by using SSIS (SQL Server Integration Services).

Acknowledgements

I would like to take this opportunity to express my deepest gratitude to my advisor Dr. Hiral Shah, Associate Professor at St. Cloud state University for her continuous encouragement and guidance throughout my project.

I express my sincere gratitude to Dr. Ben Baliga, Professor and Graduate Director for Engineering Management Program at St. Cloud State University for his guidance of my coursework and project.

I would like to thank Prof. Balsy Kasi for serving on the committee and support throughout the project.

Table of Contents

Page

List of Table 6

List of Figures 7

Chapter

I. Introduction 9

Introduction 9

Problem Statement 9

Nature and Significance of the Problem 10

Objective of the Project 11

Project Questions/Hypotheses 11

Definition of Terms 11

Summary 12

II. Background and Review of Literature 13

Introduction 13

Background Related to the Problem 13

Literature Related to the Problem 13

Literature Related to the Methodology 15

Summary 22

III. Methodology 23

Introduction 23

Chapter Page

Design of the Study 23

Data Collection 26

Data Analysis 26

Budget 26

Timeline 27

Summary 27

IV. Data Presentation and Analysis 28

Introduction 28

Data Presentation 28

Data Analysis 44

Summary 45

V. Results, Conclusion, and Recommendations 46

Introduction 46

Results 46

Conclusion 47

Recommendations 47

References 48

List of Table

Table Page

1. Project timeline 27

List of Figures

Figure Page

1. SLDC process 16
2. Project flow 24
3. CCM and CLAIMS tables 29
4. Creating a new project 30
5. Creating a new SSIS package 30
6. Adding data flow task to the package 31
7. Adding sources to the data flow 32
8. Source connection manager 33
9. OLEDB source configuration for CCM data 34
10. Column mapping of OLEDB source for CCM data 35
11. Source configuration for CLAIMS data 36
12. Column mapping for CLAIMS data 37
13. Sorting Case_ID column for CCM data 38
14. Sorting Case_ID column for CLAIMS data 39
15. Renaming sort tasks 39
16. Input output select pop up for merge Join task 40
17. Merge Join task configuration 41
18. OLEDB destination configuration 42
19. Mappings for OLEDB destination 43
20. Final package 43

Figure Page

21. Merged data table in SQL platform 44

Chapter I: Introduction

Introduction

The project is being implemented at one of banking systems which was in the process of introducing a new online banking functionality that will enable customers to perform wire transfers by logging into online banking. The first customer wave of this functionality was deployed in October 2015 for customers in NC, SC, WA, and NM. It will be followed by a second wave of deployment across the remaining regions/markets in May 2016.

Initially company was unable to link the suspicious wire transfers data with confirmed fraud wire transfers data. They needed a tool which helps them in integrating such data from various applications. This was achieved by using an ETL tool.

SQL Server Integration Services (SSIS) is one of the ETL tools that performs ETL operations; i.e. Extract, Transform and Load data. It aids in combining data from different sources stored in different storage systems. SSIS is responsible for connecting to each data source, extracting the data and merging it into a single dataset. Different parts of business use different storage technologies or different schemas to represent the same data. In these cases, SSIS performs the homogenization of the information.

Problem Statement

Initially there was no linkage to connect CCM (Customer Content Management) with CLAIMS (Claims Loss Accounting Investigation Management

System) data to obtain the claims resolution information. As a result, the business was unable to report wire loss information accurately.

Nature and Significance of the Problem

Confirmed Fraudulent claims associated with online wire transactions are logged manually in both CCM (Customer Content Management) and CLAIMS (Claims Loss Accounting Investigation Management System) applications. Claims in CCM and CLAIMS are handled in the following manner:

1. A case is opened in CCM for online wire transactions identified as either suspicious or confirmed fraud by the virtual fraud channel team.
2. Once the case is opened in CCM, an outbound call is placed to the customer to confirm that the suspicious transactions identified by the virtual fraud channels team are in fact fraudulent.
3. Only for the transactions confirmed as fraud is a case opened in CLAIMS for resolution.

Provisioning the CCM wire data that contains the linkage to CLAIMS will enable the business to accurately report the losses associated with online wire transfer. Once the data is integrated and analysis is performed it helps in understanding the current situation of business and helps in making effective decisions.

Objective of the Project

To source the claims associated with online wire transactions from the CCM application and provision it in the SQL Server platform so further Data analysis can be performed to help the business in making effective decisions.

Project Questions/Hypotheses

The following questions have been answered in this project:

- What is the efficient way to integrate data?
- How is the data consistency achieved?
- Can the project meet Go Live date?

Definition of Terms

SQL: Structured Query Language is a standard interactive and programming language is used to communicate with a database by retrieving and updating the data in it.

ETL: Stands for Extract, Transform and Load helps in extracting, transforming and loading data from one source to other.

SSDT: SQL Server Data Tools is a development tool that helps in building Integration Services packages, Analysis Services data models, and Reporting Services reports.

SSIS: SQL Server Integration Services is a component of the Microsoft SQL Server database software that can be used to perform a broad range of data migration tasks. It is one of the many available ETL tools in the market.

SSRS: SQL Server Reporting Services (SSRS) is a server-based report generating software system from Microsoft. It is part of suite of Microsoft SQL Server services, including SSAS (SQL Server Analysis Services) and SSIS.

Summary

This chapter gives us the brief introduction to the project and helps in understanding the problem as why this project is taken up and the objective of this project.

.

Chapter II: Background and Review of Literature

Introduction

This chapter focuses on the understanding the background related to problem and reviewing the literature related to problem and Methodology.

Background Related to the Problem

Client is one of the banking systems which has recently provided the customers with online wire transfer facility to its customers in few parts of USA. Since it is an online wire transfer it is important for the client to be aware of the fraud cases and up to what extent they are being solved else there will be no measure on how their business is running. Initially the data is being entered into each of the applications CCM and CLAIMS manually by the customer care people but as there is no linkage to both the applications it has been difficult for the business teams to estimate as how many cases are being reported and how many are solved. Calculating them manually every time is not practical and may not be accurate and also it requires lot of labor. So the client needed an automated application as a onetime investment which they can use in future. This project helps in understanding how that can be achieved.

Literature Related to the Problem

With the increasing business needs to manage large amounts of data; data is typically stored in many different data storage systems, and extracting data from all sources and merging the data into a single, consistent dataset that can later be used for data analysis is challenging [1]. Integration Services can connect to a wide variety of data sources, including multiple sources in a single package. A package can connect to relational databases by using .NET and OLE DB providers, and to many legacy databases by using ODBC drivers. It can also connect to flat files, Excel files, and Analysis Services projects [2].

Integration Services includes source components that perform the work of extracting data from flat files, Excel spreadsheets, XML documents, and tables and views in relational databases from the data source to which the package connects. Next, the data is typically transformed by using the transformations that Integration Services includes. After the data is transformed to compatible formats, it can be merged physically into one dataset. After the data is merged successfully and transformations are applied to data, the data is usually loaded into one or more destinations. Integration Services includes destination for loading data into flat files, raw files, and relational databases. The data can also be loaded into an in-memory record set and accessed by other package elements [3]. Moreover as the requirements start increasing, tracing the information can be quite difficult when scripts are used but it turns out to be quite easier in the form of workflows feasible through such integration services.

Microsoft Integration Services is a platform for building enterprise-level data integration and data transformations solutions. It helps in pulling the data from multiple sources, cleanse and transform the data, and load the data into appropriate data stores for analysis and reporting. Its functionalities include:

- ETL services–Extract, Transform and load data.
- Merge data from multiple sources to single destination.
- Connection to different types of source and destination like flat files, excel sheets, XML files, ODBC and OLEDB.
- Merge data from heterogeneous data sources into SQL Server [4].

Literature Related to the Methodology

SLDC**:** SDLC stands for Software Development Life Cycle. SDLC is the process consisting of a series of planned activities to develop or alter the software products. The SDLC aims to produce a high quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates [5].

The following figure (Figure 1) is a graphical representation of the various stages of SDLC.

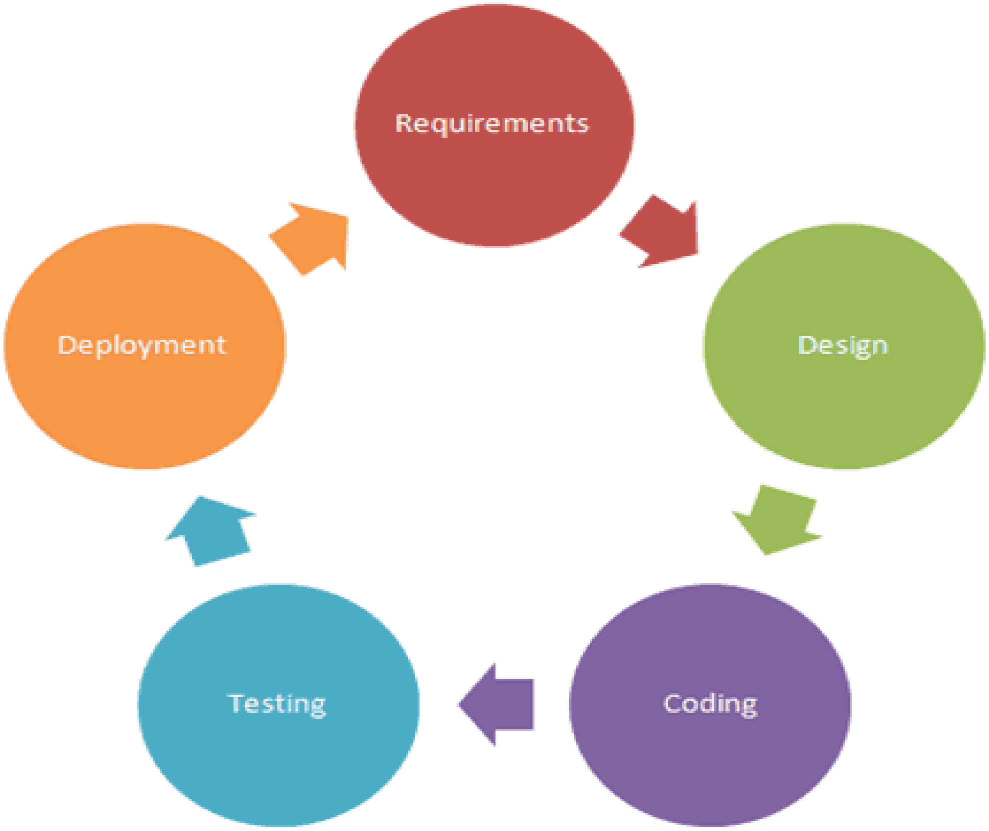


Figure 1. SLDC process [6].

There are typically five phases in SLDC process:

Phase 1: Requirements Gathering/Analysis*:* This phase is critical to the success of the project. Expectations (whether of a client or your team) need to be fleshed out in great detail and documented. This is an iterative process with much communication taking place between stakeholders, end users and the project team. The following techniques can be used to gather requirements:

- Identify and capture stakeholder requirements using customer interviews and surveys.

- Build multiple use cases to describe each action that a user will take in the new system.
- Prototypes can be built to show the client what the end product will look like.

In a corporate setting, this means taking a look at your customers, figuring out what they want, and then designing what a successful outcome would look like in a new bit of software [5].

Phase 2: Design: Technical design requirements are prepared in this phase by lead development staff that can include architects and lead developers. The Business Requirements are used to define how the application will be written. Technical requirements will detail database tables to be added, new transactions to be defined, security processes and hardware and system requirements [5].

Some of the activities involved in this stage:

Risk analysis: Threats and vulnerabilities which may arise from interactions with other systems.

- External or legacy code needs to be analyzed to determine if there are security vulnerabilities.
- High-risk privacy projects could require review with a legal department. This review should consider what personal data to collect, how to collect it, and permissions/authorizations to make changes. This type of review is especially necessary with corporate projects.

Functional specifications:

- Includes a description of interface requirements such as definition of data entry fields (allow numeric or alpha only, can it be left blank)?
- Important details, like: can date entered be before current date? What time zone will user logins default to?
- Workflow–after clicking approve button, which screen appears next?
- Audit trail for every update on the database. This is where error monitoring and logging tools can be useful.

Non-functional specifications:

- Extensibility of the system–will current system easily allow new enhancements or features with the next rollout? This is critical for any application that you'll be adding new features and updating often.
- Has the current or future capacity been analyzed for database requirements? Will the current build plan result in capacity issues shortly after you finish building?
- Performance and response time–Has the expected response time been determined?
- Resource Constraints–Are there constraints that need to be taken into consideration in this phase? Common ones include disk space, bandwidth, etc. [5].

Phase 3: Coding: This phase is the actual coding and unit testing of the process by the development team. After each stage, the developer may demonstrate the work accomplished to the Business Analysts and tweaks and enhancements may be required. It's important in this phase for developers to be open-minded and flexible if any changes are introduced. This is normally the longest phase of the SDLC. The finished product here is input to the Testing phase [5].

Phase 4: Testing: Once the application is migrated to a test environment, different types of testing will be performed including integration and system testing. User acceptance testing is the last part of testing and is performed by the end users to ensure the system meets their expectations. At this point, defects may be found and more work may be required in the analysis, design or coding. Once sign-off is obtained by all relevant parties, implementation and deployment can begin [5].

Phase 5: Implementation/Deployment: The size of the project will determine the complexity of the deployment. Training may be required for end users, operations and on-call IT staff. Roll-out of the system may be performed in stages starting with one branch then slowly adding all locations or it could be a full blown implementation [5].

One of two methods can be followed in a SDLC process. Waterfall is the more traditional model and has a well-structured plan and requirements to be followed. This method works well for large projects that may take many months to develop. The Agile Methodology is more flexible in the requirements, design and coding process and is very iterative. This process works best for smaller projects and expectations of continuous improvement to the application. Whether you use one over the other will also depend to a large extent on the corporation and skills of the IT dept. [5].

Agile Methodology: Agile is a software development methodology to build a software incrementally using short iterations of 1 to 4 weeks so that the development process is aligned with the changing business needs. Instead of a single-pass development of 6 to 18 months where all the requirements and risks are predicted upfront, Agile adopts a process of frequent feedback where a workable product is delivered after 1 to 4 week iteration [7].

Agile Manifesto principles include:

- Customer Satisfaction−Highest priority is given to satisfy the requirements of customers through early and continuous delivery of valuable software.
- Welcome Change−Changes are inevitable during software development. Ever-changing requirements should be welcome, even late in the development phase. Agile processes should work to increase customers' competitive advantage.
- Deliver a Working Software−Deliver a working software frequently, ranging from a few weeks to a few months, considering shorter time-scale.
- Collaboration−Business people and developers must work together during the entire life of a project.
- Motivation−Projects should be built around motivated individuals. Provide an environment to support individual team members and trust them so as to make them feel responsible to get the job done.

- Face-to-face Conversation−Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.
- Measure the Progress as per the Working Software−Working software is the key and it should be the primary measure of progress.
- Maintain Constant Pace−Agile processes aim towards sustainable development. The business, the developers, and the users should be able to maintain a constant pace with the project.
- Monitoring−Pay regular attention to technical excellence and good design to enhance agility.
- Simplicity−Keep things simple and use simple terms to measure the work that is not completed.
- Self-organized Teams−an agile team should be self-organized and should not depend heavily on other teams because the best architectures, requirements, and designs emerge from self-organized teams.
- Review the Work Regularly−Review the work done at regular intervals so that the team can reflect on how to become more effective and adjust its behavior accordingly [8].

The Agile Method ensures that value is optimized throughout the development process. The use of iterative planning and feedback results in teams that can continuously align a delivered product that reflects the desired needs of a client. It easily adapts to changing requirements throughout the process by measuring and evaluating the status of a project. The measuring and evaluating allows accurate and early visibility into the progress of each project. This method offers a light framework for assisting teams. It helps them function and maintain focus on rapid delivery. This focus assists capable organizations in reducing the overall risks associated with software development [9].

Summary

This chapter was mainly focused on describing the background related to the problem, and an in depth details of the literature related to the problem. Also, the background literature review related to the methodology of the project has been explained in a detailed manner. In the next chapters, the frame work of the project, detailed phases of implementations, budget, project timeline and their advantages will be listed.

Chapter III: Methodology

Introduction

This chapter focuses mainly on explaining the procedure that has been used for solving the problem and also understand about the methodology that has been used to achieve the solution for the problem. Budget and timeline time line involved to conduct the entire project is also specified.

Design of the Study

This project was conducted in four phases:

Phase 1: Requirement Gathering and Documentation: The business team has gathered all the requirements based upon the client and end user perspective. Once the requirements are gathered, functional and technical documents were prepared by the respective teams. Functional document includes business related information where as technical document includes technical specifications for the project.

Phase 2: Design and Development: Once the functional and technical documents are ready the developers were set to build the code.

Project flow is as shown and described below for a clear understanding (Figure 2).

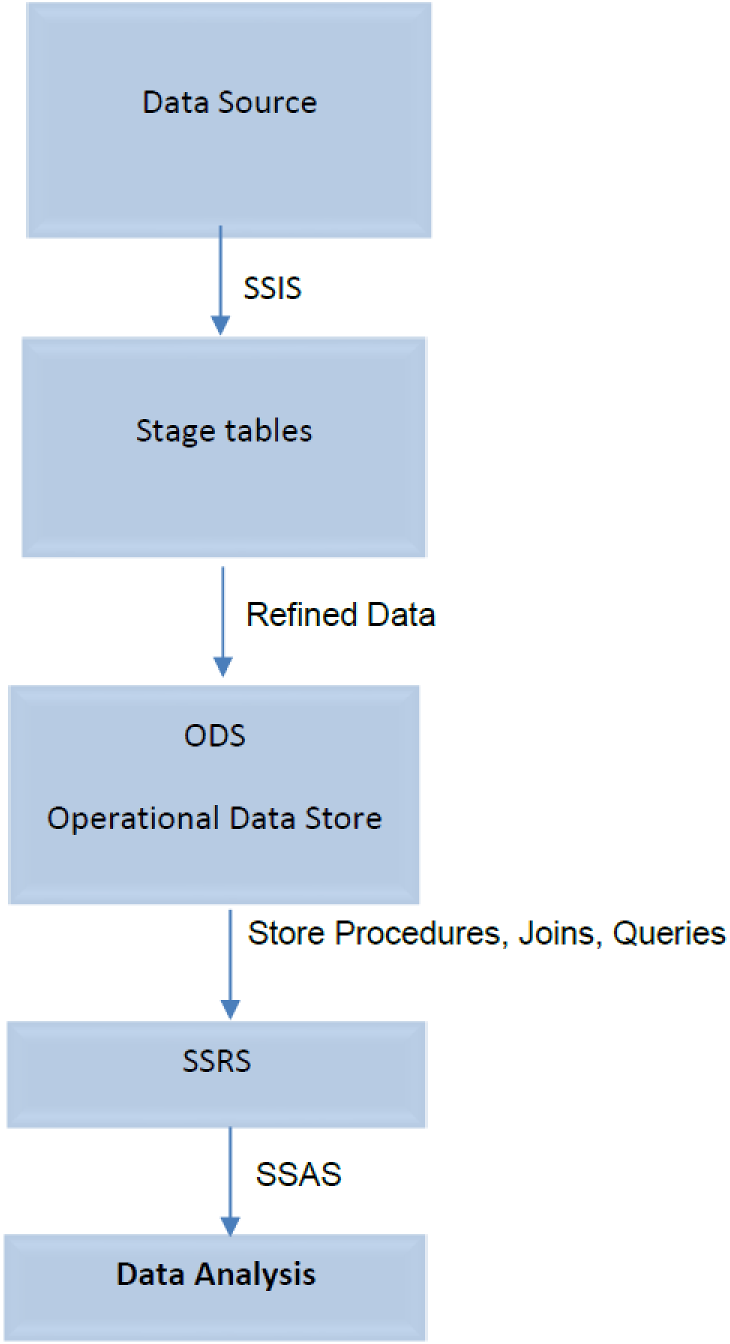


Figure 2. Project flow.

- Data is pulled from the source(s), integrated using SSIS and stored in the staging tables and from the staging tables, refined data is pulled and stored in ODS (Operational Data Storage).
- From this ODS, required data depending on the requirement is pulled using store procedures to create reports and from the reports data analysis will be performed.

Since this project is focused on performing the data integration and all remaining stages be taken care by corresponding teams, the main tasks for the development team were:

- Creating SSIS packages to move the source data into the staging tables.
- Mapping the columns from source table and destination table with appropriate data conversions.
- Configuring various components like Connection Manager, Control Flow and Data Flow.

Testing: Once the code is developed, it was tested within both QA (Quality Assurance) and UAT (User Acceptance Testing) environments to ensure there are no possible effective bugs. Once the issues are detected they were resolved immediately.

Deployment and production: After the integration and associated testing is performed; sign off and approval is received from the client and the code is deployed and is deemed ready for production. In the production environment, as soon as an issue arises it was addressed by the production team by updating the code with a newer version of code and with a configuration change.

Data Collection

Data collection for this project is carried out in the following manner:

- Data is stored in the form of tables in SQL Databases.
- Sources are identified from where the data is to be integrated.
- SSIS packages are created by using the specified naming format and store in the defined location.
- Required staging tables are created once the data is integrated from different sources.
- Ensuring that the tables have the same column names as in the source tables and same datatypes and lengths.
- Creating new columns if required using the same naming format.
- Ensuring that the developers are given proper permissions to these tables.

Data Analysis

Once the CCM data is integrated with CLAIMS data, data analysis is made using SSRS (Sql Server Reporting Services). These reports are generated using the input parameters and sent to business teams for analyzing how many claims are being reported and how many of them are closed.

Budget

The entire costs for the resources, software installation and licenses and other miscellaneous costs were covered by the client.

Timeline

Table 1 shows the various stages of the project.

Table 1. Project timeline.

Stages of the project	TimeLine
Requirement Gathering	November 2015
Documentation	December 2015- January 2015
Design and Development	February 2016- April 2016
Testing	May2016-June 2016
Final Deployment	June 2016
Production	June 2016
Final Defense	July 2016

Summary

This chapter helped in understanding the project flow and framework in a very detailed manner. Tools used in this project are discussed in this chapter and also the project timeline. Following the steps carried out from this chapter we can see how data is analyzed in the further chapter.

Chapter IV: Data Presentation and Analysis

Introduction

This chapter helps in understanding how the data collected is presented and how it is analyzed.

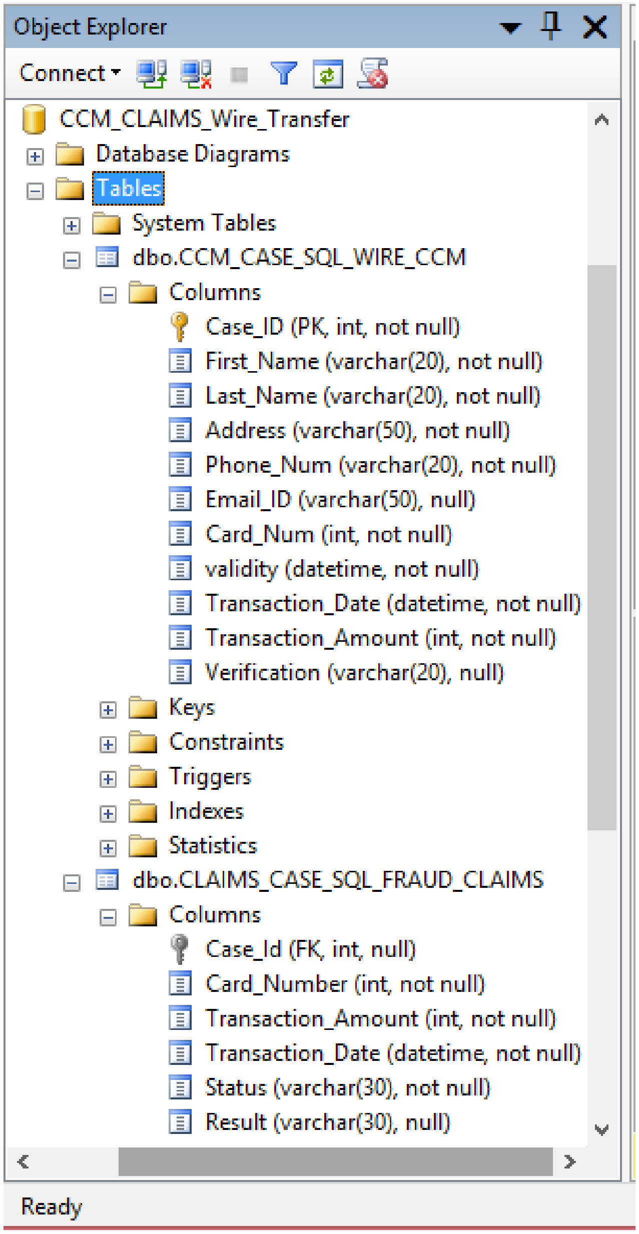
Data Presentation

Once data is collected and stored in Sql database (CCM_CLAIMS_Wire_ Transfer) in the form of tables, it is extracted from the tables through SSIS and then integrated. The data from both CCM and CLAIMS applications were stored in these two tables respectively.

- CCM application data- dbo.CCM_CASE_SQL_WIRE_CCM
- CLAIMS application data- dbo.CLAIMS_CASE_SQL_FRAUD_CLAIMS

There is a column in dbo.CCM_CASE_SQL_WIRE_CCM table, Case_ID which acts a primary key to uniquely identify each row in the table. The Case_ID column in dbo.CLAIMS_CASE_SQL_FRAUD_CLAIMS is the foreign key which references the Case_ID column in dbo.CCM_CASE_SQL_WIRE_CCM table.

The below figure (Figure 3) shows the columns of each table:

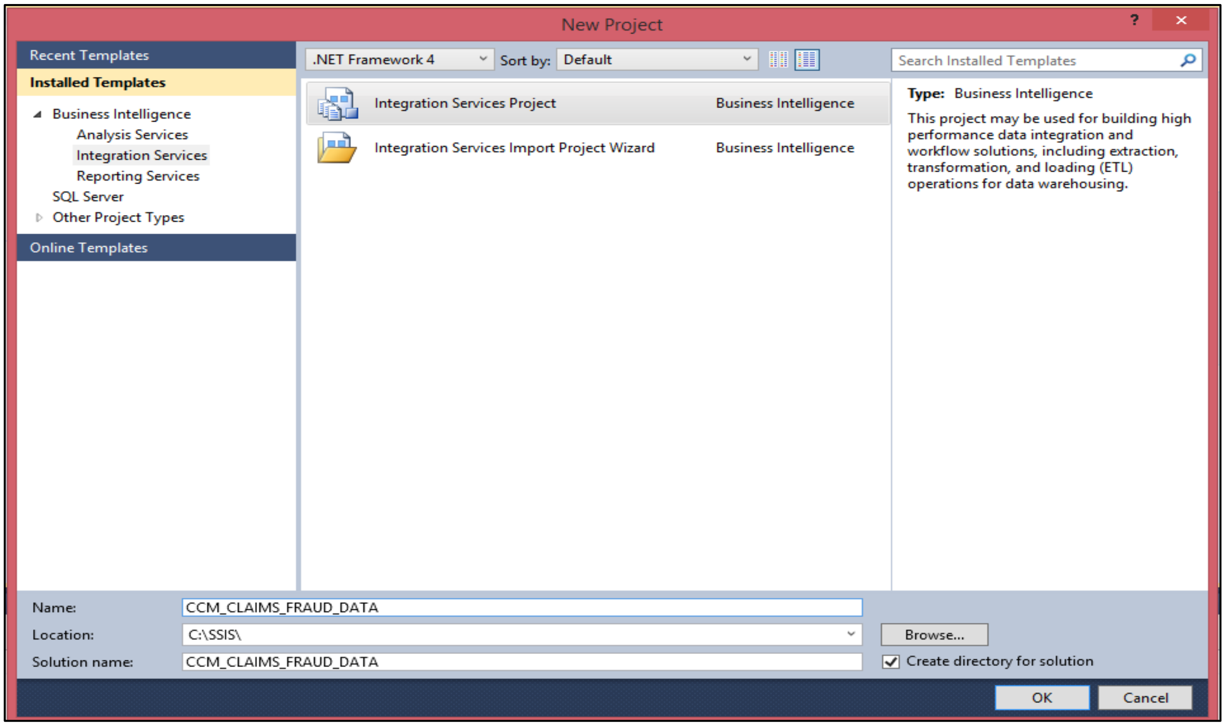


```
CCM_CLAIMS_Wire_Transfer
  Database Diagrams
  Tables
    System Tables
    dbo.CCM_CASE_SQL_WIRE_CCM
      Columns
          Case_ID (PK, int, not null)
          First_Name (varchar(20), not null)
          Last_Name (varchar(20), not null)
          Address (varchar(50), not null)
          Phone_Num (varchar(20), not null)
          Email_ID (varchar(50), null)
          Card_Num (int, not null)
          validity (datetime, not null)
          Transaction_Date (datetime, not null)
          Transaction_Amount (int, not null)
          Verification (varchar(20), null)
      Keys
      Constraints
      Triggers
      Indexes
      Statistics
    dbo.CLAIMS_CASE_SQL_FRAUD_CLAIMS
      Columns
          Case_Id (FK, int, null)
          Card_Number (int, not null)
          Transaction_Amount (int, not null)
          Transaction_Date (datetime, not null)
          Status (varchar(30), not null)
          Result (varchar(30), null)
```

Figure 3. CCM and CLAIMS tables.

Since the data is in two different tables it was necessary to integrate them into one table to understand the data clearly and make estimates. The basic steps included in integrating data are shown below for brief understanding.

***Step 1*: *Create new SSIS package*.** SQL Server Data tools is launched and a new project and solution is created and then a new package is added to the project. This is shown in the following figures (Figures 4 and 5).



Type: Business Intelligence

This project may be used for building high performance data integration and workflow solutions, including extraction, transformation, and loading (ETL) operations for data warehousing.

Name: CCM_CLAIMS_FRAUD_DATA

Location: C:\SSIS\

Solution name: CCM_CLAIMS_FRAUD_DATA

☑ Create directory for solution

Figure 4. Creating a new project.

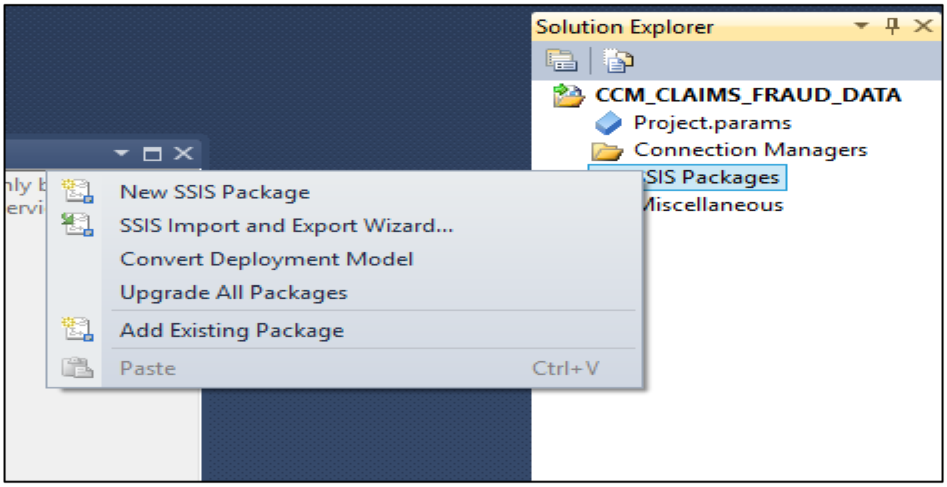
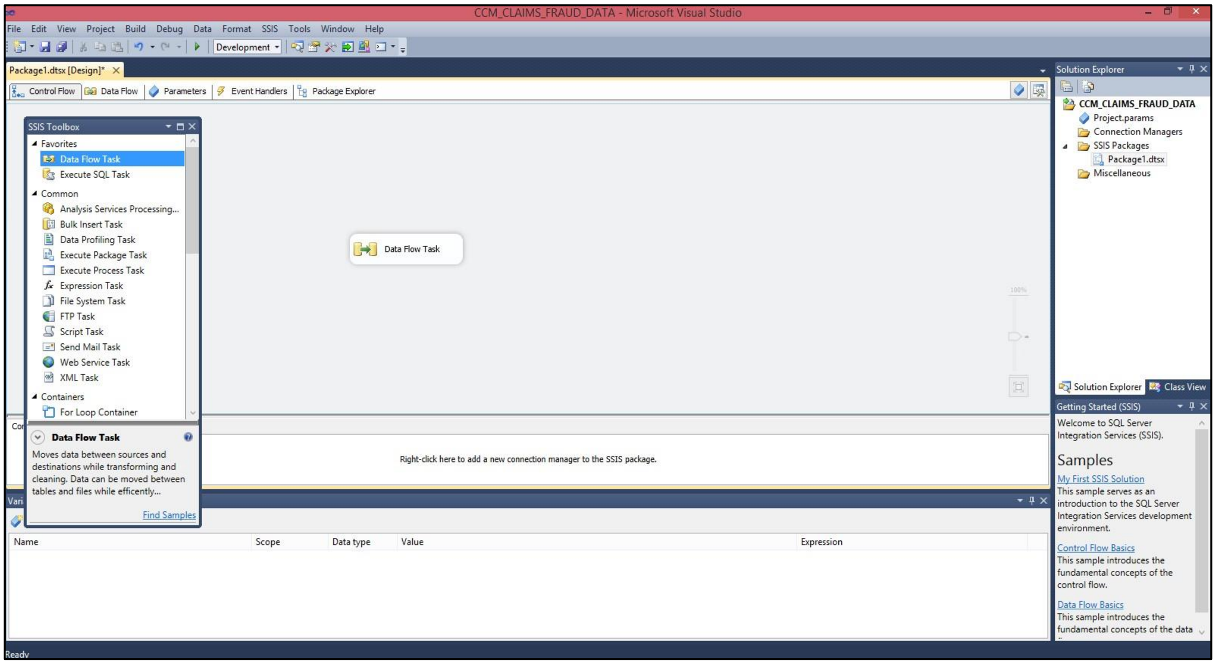


Figure 5. Creating a new SSIS package.

Step 2: Adding a data flow task. Since data is being transferred to a new destination from the sources data flow task is added to the control flow panel to help the data flow from source to destination. Data flow task was added by dragging and dropping it from the tool box. Figure 6 shows data flow task added to the control flow panel.



Moves data between sources and destinations while transforming and cleaning. Data can be moved between tables and files while efficently...

Right-click here to add a new connection manager to the SSIS package.

Welcome to SQL Server Integration Services (SSIS).

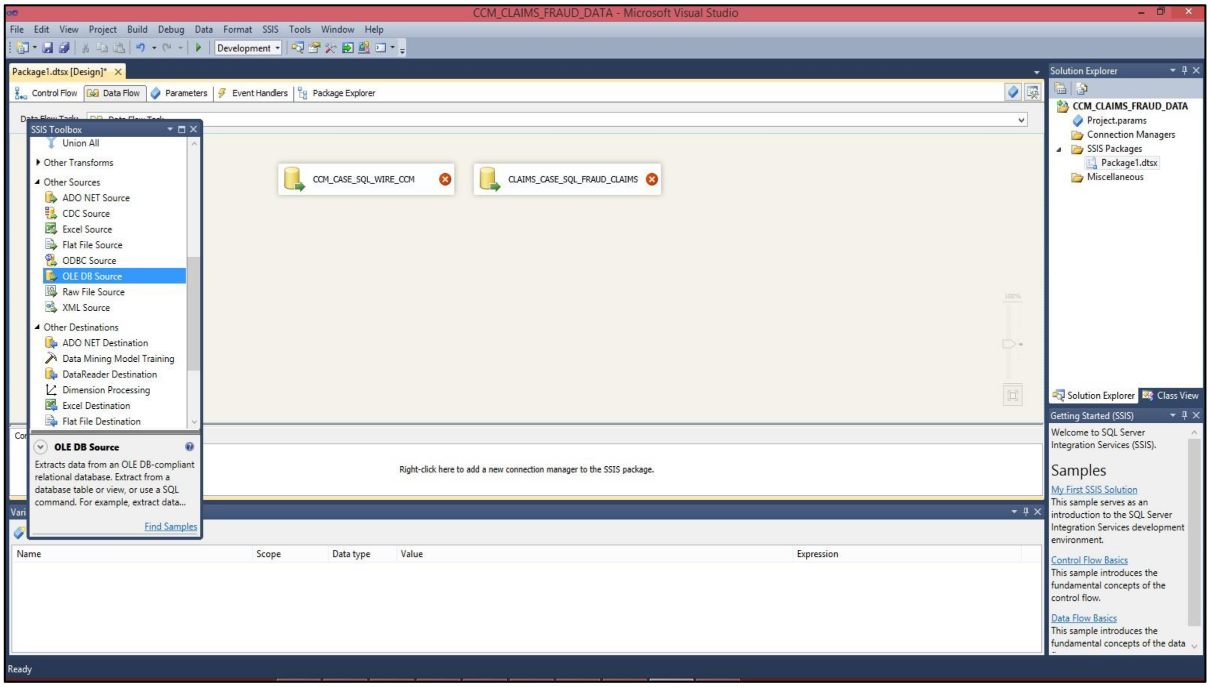
This sample serves as an introduction to the SQL Server Integration Services development environment.

This sample introduces the fundamental concepts of the control flow.

This sample introduces the fundamental concepts of the data

Figure 6. Adding data flow task to the package.

Step 3: Adding sources*.* Since data was integrated from SQL tables, both sources were OLEDB Source. Since data is integrated from two table two sources were added to the data flow panel. Once both the sources are added they were renamed with the corresponding tables to avoid confusion. The same is represented in the figure below (Figure 7).



OLE DB Source

Extracts data from an OLE DB-compliant relational database. Extract from a database table or view, or use a SQL command. For example, extract data...

Right-click here to add a new connection manager to the SSIS package.

Welcome to SQL Server Integration Services (SSIS).

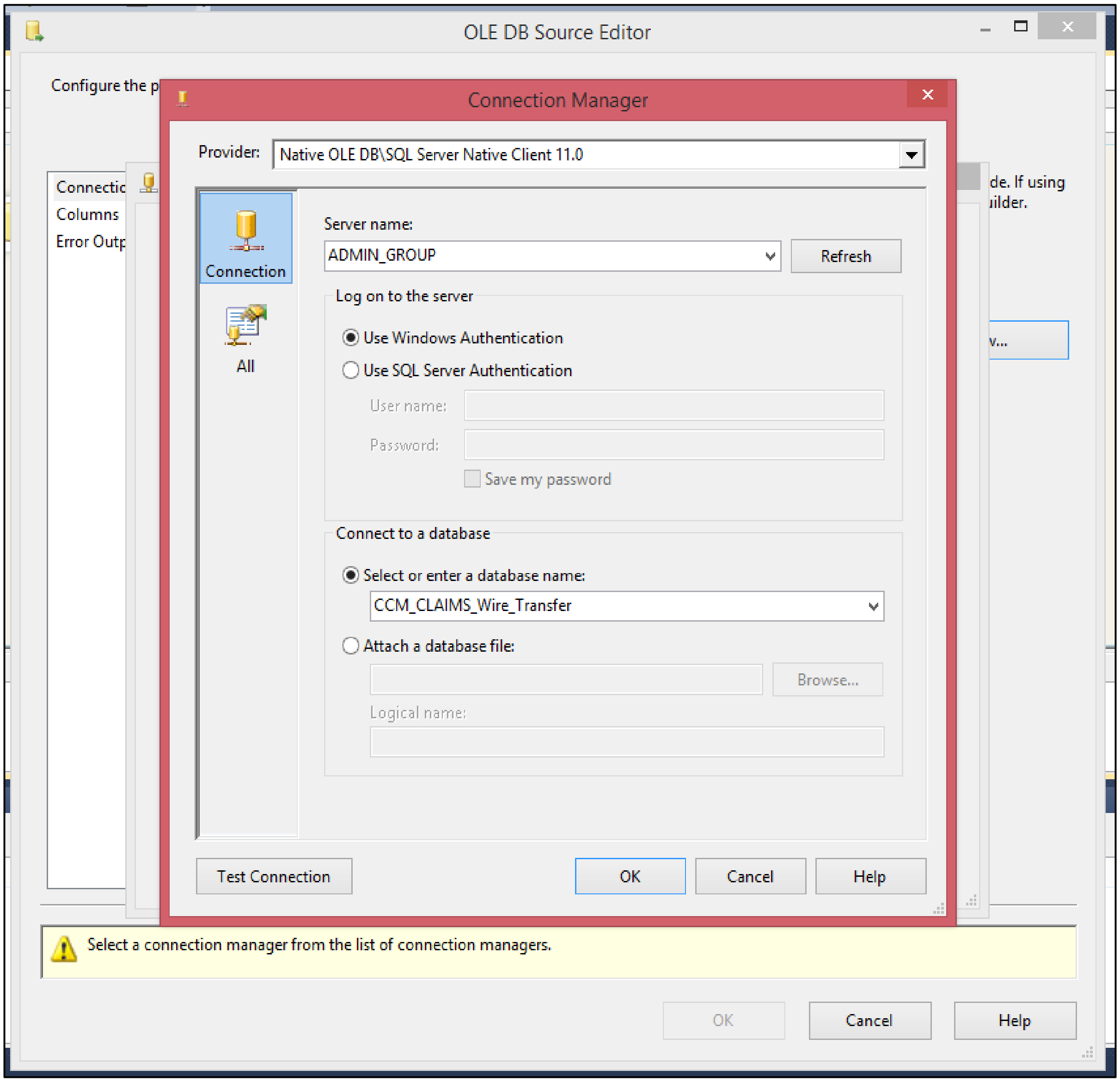
This sample serves as an introduction to the SQL Server Integration Services development environment.

This sample introduces the fundamental concepts of the control flow.

This sample introduces the fundamental concepts of the data

Figure 7. Adding sources to the data flow.

Step 4*: *Configuring sources. A new connection manager is created for the sources to the database where the required tables are stored. This helps the OLEDB sources to get connected to the database in SQL Server. Once the connection is established test connection is clicked to ensure whether a connection is established successfully. Next, each of the OLEDB sources were configured with the corresponding tables using the connection manager. The same is represented in the figure below (Figure 8).



OLE DB Source Editor

Connection Manager

Provider: Native OLE DB\SQL Server Native Client 11.0

Server name: ADMIN_GROUP

Log on to the server

● Use Windows Authentication

○ Use SQL Server Authentication

User name:

Password:

□ Save my password

Connect to a database

● Select or enter a database name:

CCM_CLAIMS_Wire_Transfer

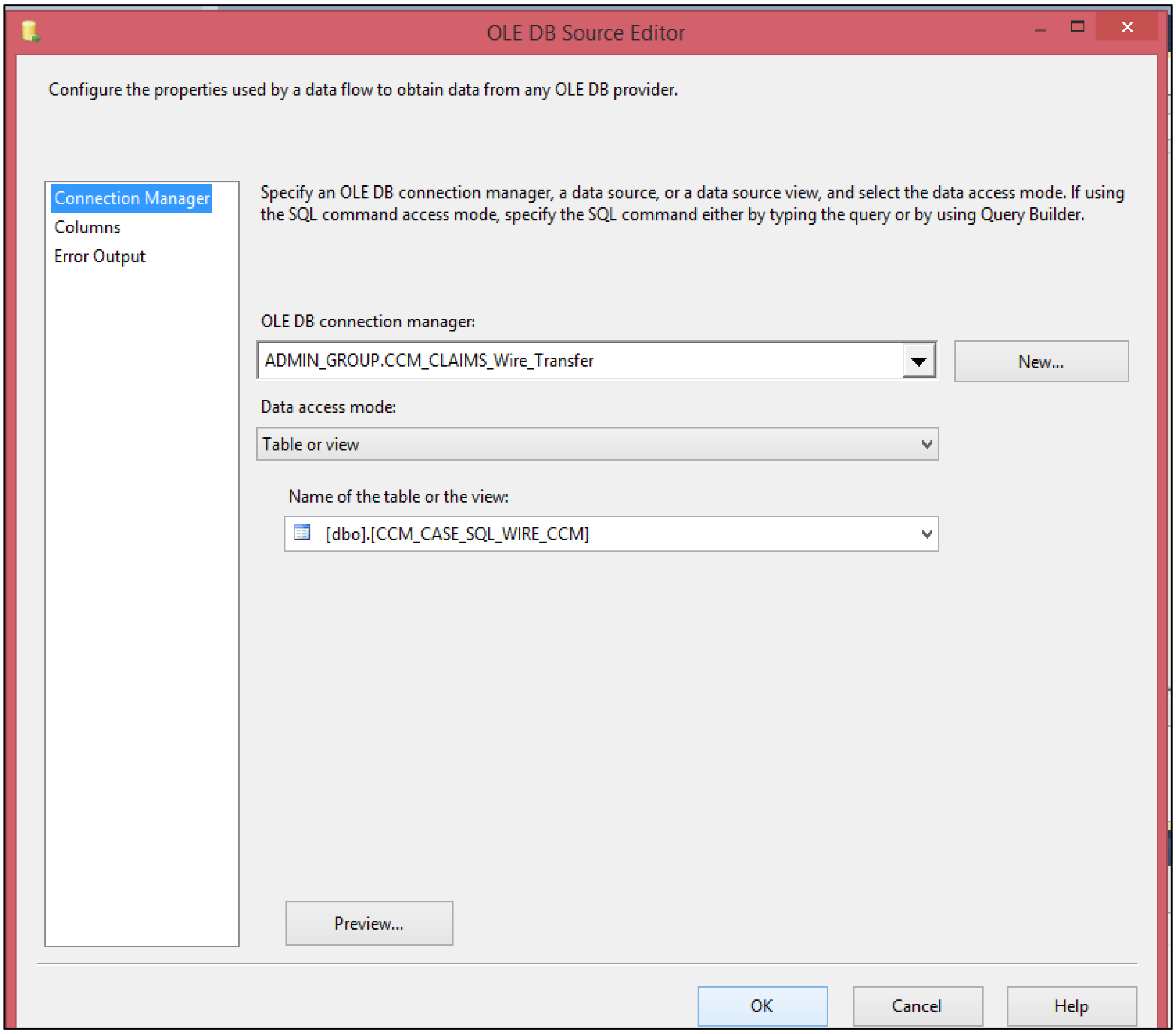
○ Attach a database file:

Logical name:

Select a connection manager from the list of connection managers.

Figure 8. Source connection manager.

Once connection is successful for source, the table which has to be connected to the source is selected. Here CCM application data table was connected to this source. This is shown in the figure below (Figure 9). Columns tab is verified to ensure that all the columns are checked. This is shown in Figure 10.



Configure the properties used by a data flow to obtain data from any OLE DB provider.

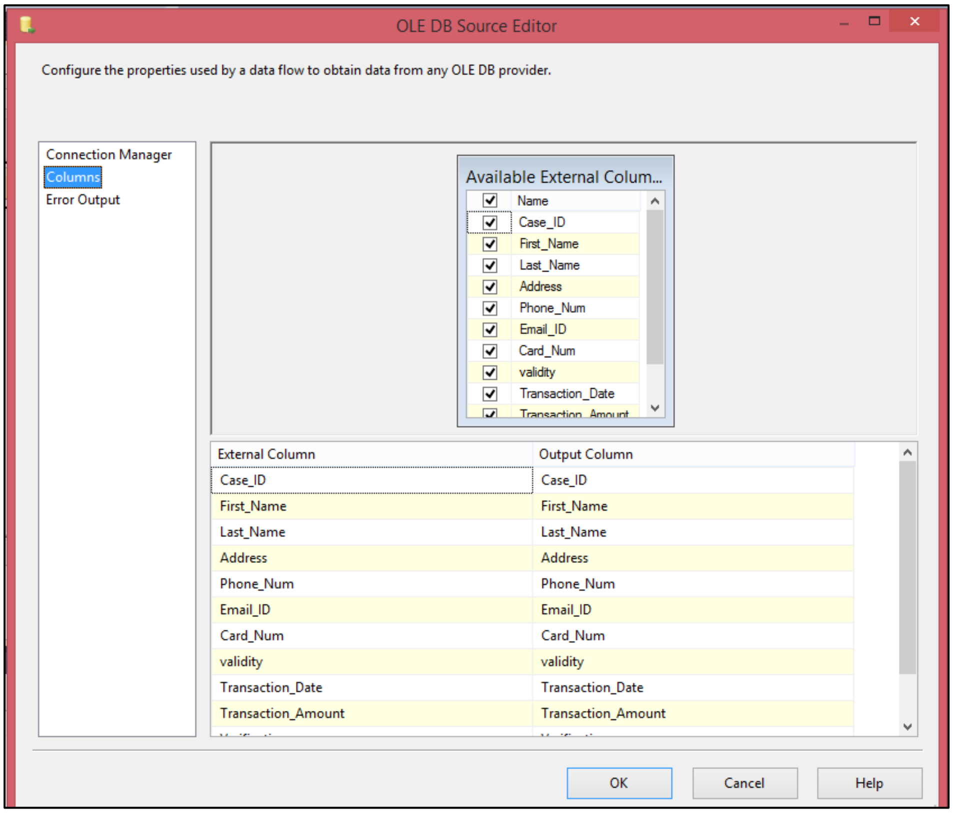
Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder.

OLE DB connection manager: ADMIN_GROUP.CCM_CLAIMS_Wire_Transfer

Data access mode: Table or view

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

Figure 9. OLEDB source configuration for CCM data.



Configure the properties used by a data flow to obtain data from any OLE DB provider.

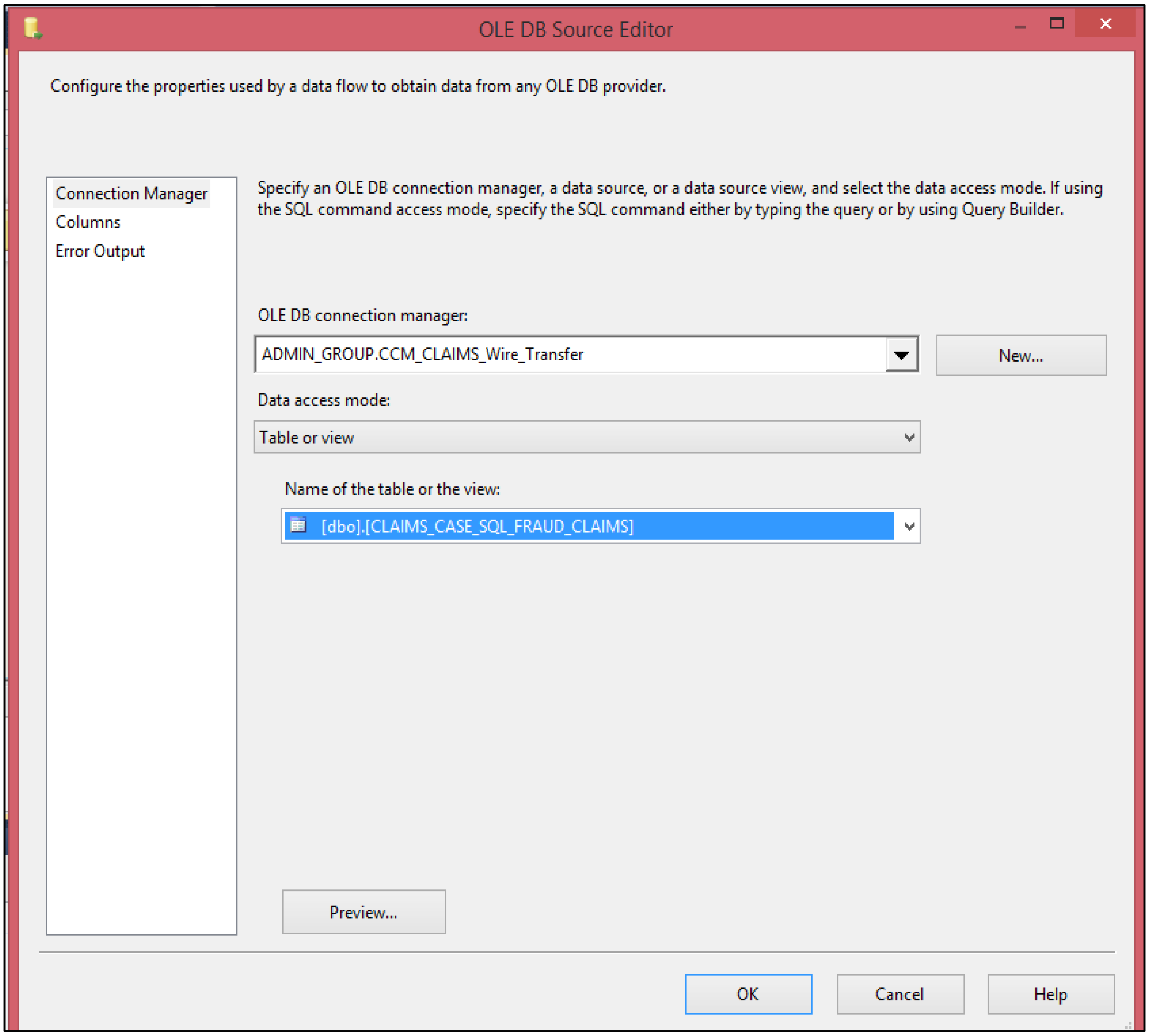
Available External Colum...

☑ Name
☑ Case_ID
☑ First_Name
☑ Last_Name
☑ Address
☑ Phone_Num
☑ Email_ID
☑ Card_Num
☑ validity
☑ Transaction_Date
☑ Transaction_Amount

External Column	Output Column
Case_ID	Case_ID
First_Name	First_Name
Last_Name	Last_Name
Address	Address
Phone_Num	Phone_Num
Email_ID	Email_ID
Card_Num	Card_Num
validity	validity
Transaction_Date	Transaction_Date
Transaction_Amount	Transaction_Amount

Figure 10. Column mapping of OLEDB source for CCM data.

Same process is repeated for the other source. Figure 11 shows the source configuration for CLAIMS data.



Configure the properties used by a data flow to obtain data from any 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.

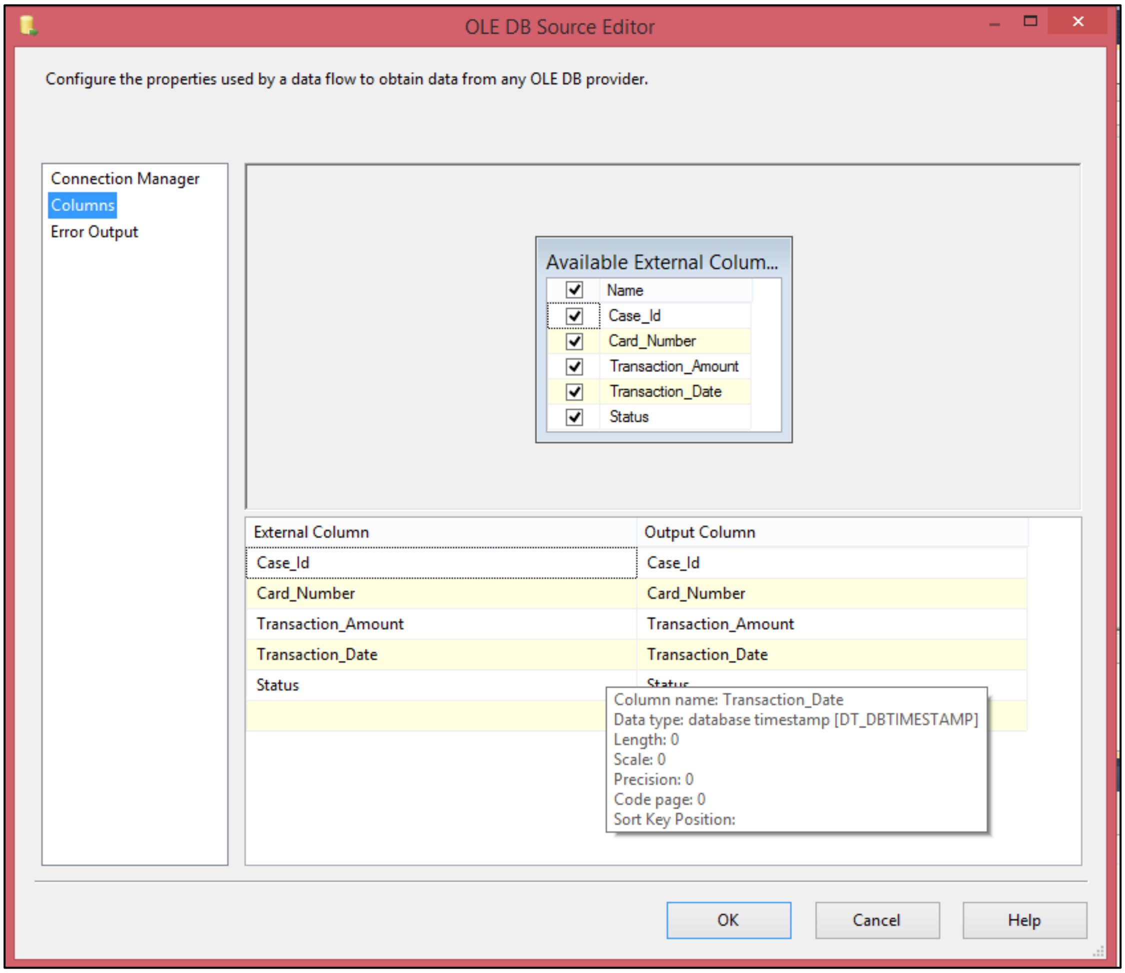
OLE DB connection manager: ADMIN_GROUP.CCM_CLAIMS_Wire_Transfer

Data access mode: Table or view

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

Figure 11. Source configuration for CLAIMS data.

The figure below (Figure 12) shows the column mapping for CLAIMS data.



Configure the properties used by a data flow to obtain data from any OLE DB provider.

Available External Colum...

☑ Name
☑ Case_Id
☑ Card_Number
☑ Transaction_Amount
☑ Transaction_Date
☑ Status

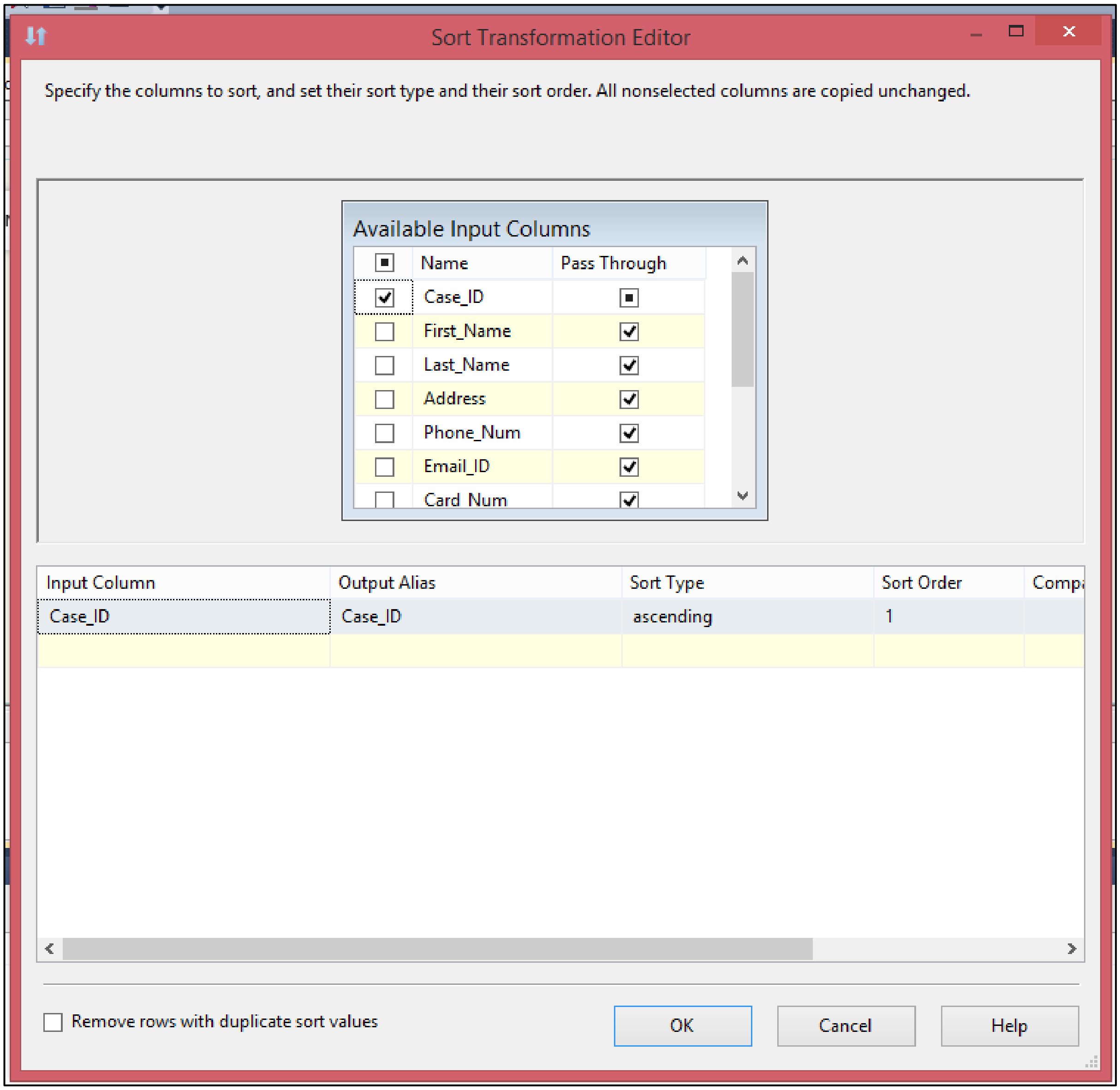
External Column	Output Column
Case_Id	Case_Id
Card_Number	Card_Number
Transaction_Amount	Transaction_Amount
Transaction_Date	Transaction_Date
Status	Status

Column name: Transaction_Date
Data type: database timestamp [DT_DBTIMESTAMP]
Length: 0
Scale: 0
Precision: 0
Code page: 0
Sort Key Position:

Figure 12. Column mapping for CLAIMS data.

Step 5: Sorting the Join column. Observing the two tables it was clear that they can be joined on the Case_ID column and hence sorting that column in advance helped in sorting the columns before head and saving time rather than sorting it while joining the tables.

Sort task is dragged and dropped under each of the sources and connected them with the blue arrow at the source and configured. The same is shown in the figure below (Figure 13) for CCM data and Figure 14 for CLAIMS data.



Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

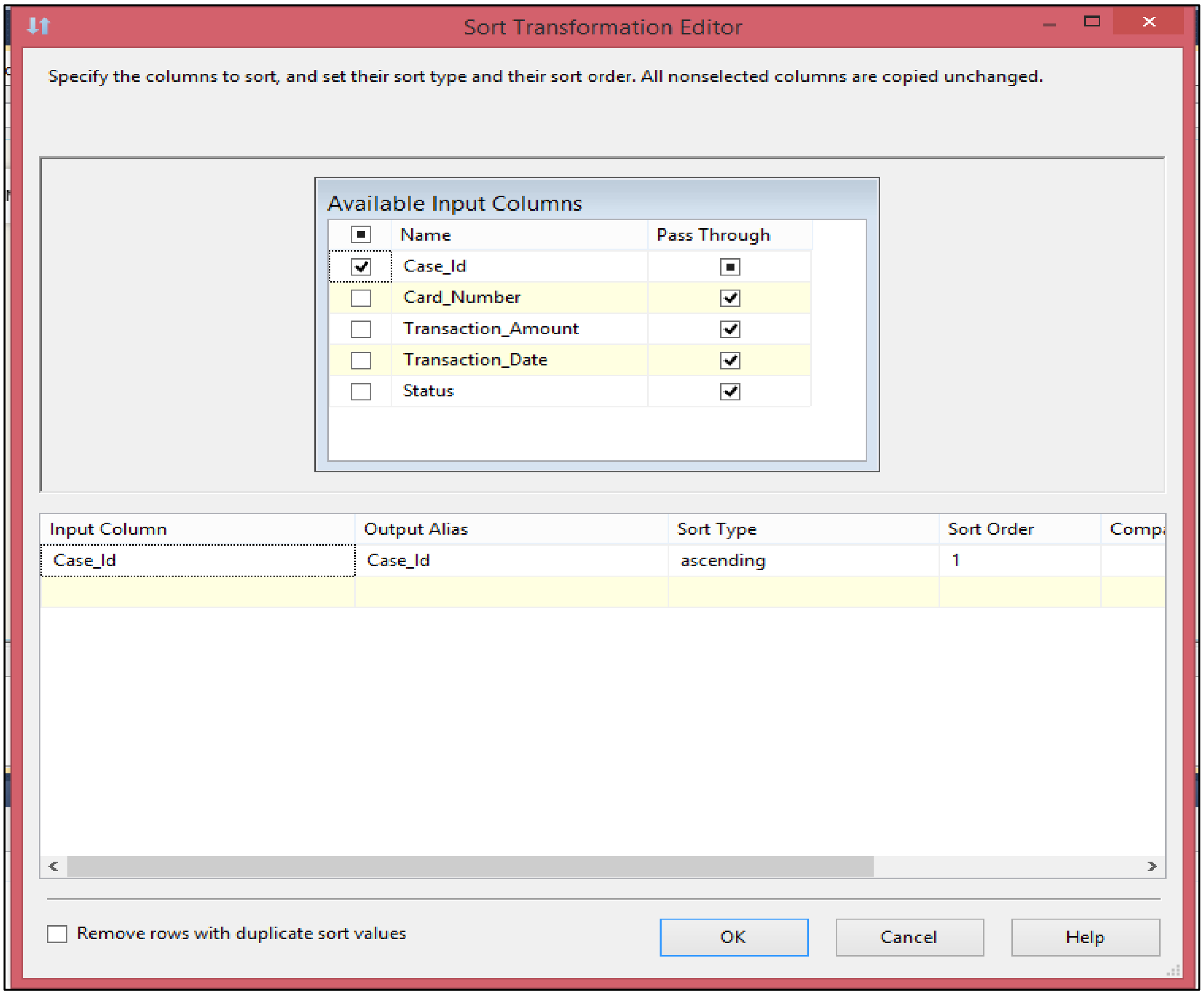
Available Input Columns

■	Name	Pass Through
☑	Case_ID	■
□	First_Name	☑
□	Last_Name	☑
□	Address	☑
□	Phone_Num	☑
□	Email_ID	☑
□	Card_Num	☑

Input Column	Output Alias	Sort Type	Sort Order	Compa
Case_ID	Case_ID	ascending	1	

□ Remove rows with duplicate sort values

Figure 13. Sorting Case_ID column for CCM data.



Sort Transformation Editor

Specify the columns to sort, and set their sort type and their sort order. All nonselected columns are copied unchanged.

Available Input Columns

■	Name	Pass Through
☑	Case_Id	■
□	Card_Number	☑
□	Transaction_Amount	☑
□	Transaction_Date	☑
□	Status	☑

Input Column	Output Alias	Sort Type	Sort Order	Comp
Case_Id	Case_Id	ascending	1	

□ Remove rows with duplicate sort values

Figure 14. Sorting Case_ID column for CLAIMS data.

Once configuration for sort task is done both the tasks are renamed to identify them easily for further reference. This is shown in the figure below (Figure 15).

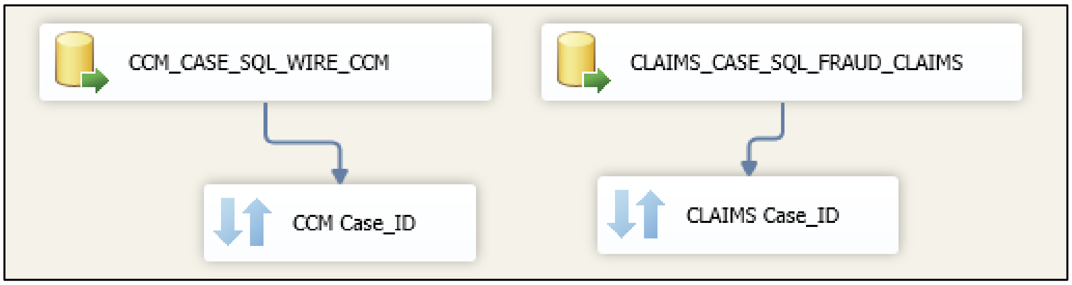
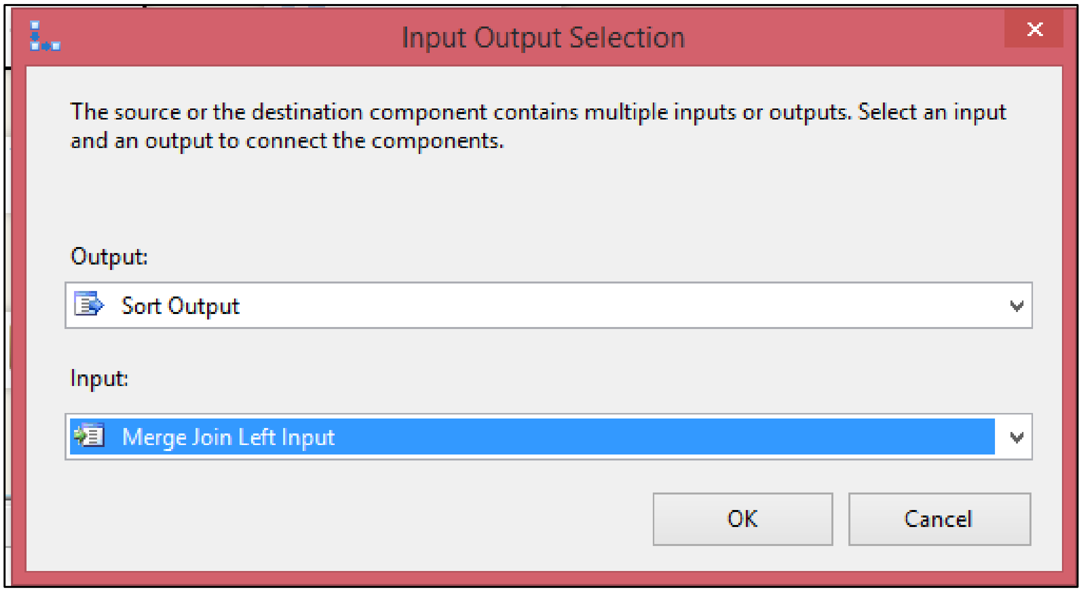


Figure 15. Renaming sort tasks.

Step 6*: *Merging the data. Since the data from both the tables must be merged by column wise they needed to be joined. To achieve this merge join task is used. Both the sorted columns are connected to the Merge Join Task. As soon as sorted column is being connected to merge join task an Input Output Select pop up appears where input selection was made. Similarly the other sorted column is connected to the Merge Join task but since the selection is already made for the first column, second column was connected automatically. Figure 16 shows the Input Output Selection pop up.



Input Output Selection

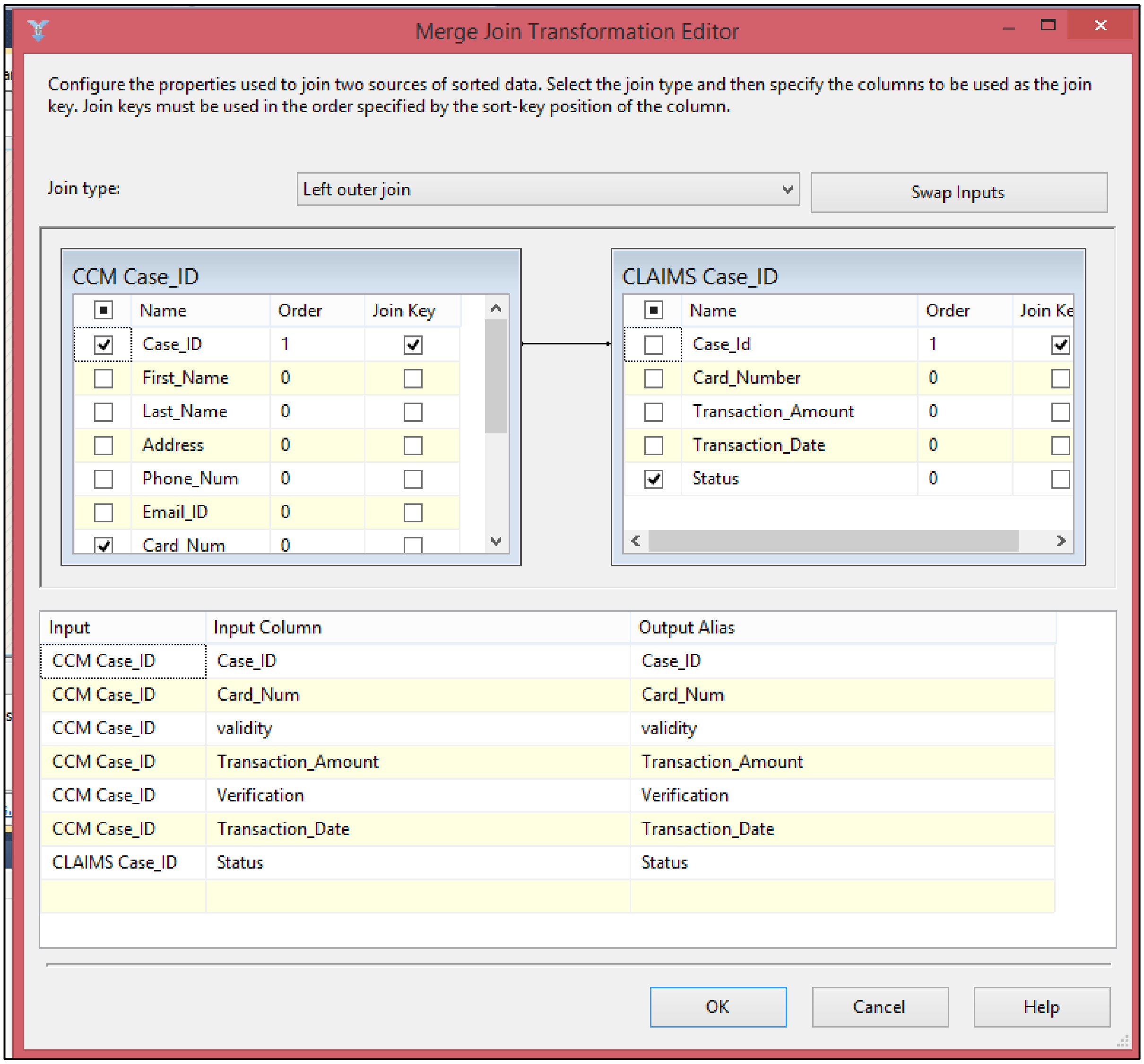
The source or the destination component contains multiple inputs or outputs. Select an input and an output to connect the components.

Output: Sort Output

Input: Merge Join Left Input

Figure 16. Input output select pop up for merge Join task.

Step 7*: *Configuring the merge Join task. Merge join task is configured by clicking on it and selecting the join type. Column selection is made based on the requirement. Figure 17 shows the Merge Join Transformation Editor where few columns are selected from both the tables.



Configure the properties used to join two sources of sorted data. Select the join type and then specify the columns to be used as the join key. Join keys must be used in the order specified by the sort-key position of the column.

Join type: Left outer join

CCM Case_ID

■	Name	Order	Join Key
☑	Case_ID	1	☑
□	First_Name	0	□
□	Last_Name	0	□
□	Address	0	□
□	Phone_Num	0	□
□	Email_ID	0	□
☑	Card_Num	0	□

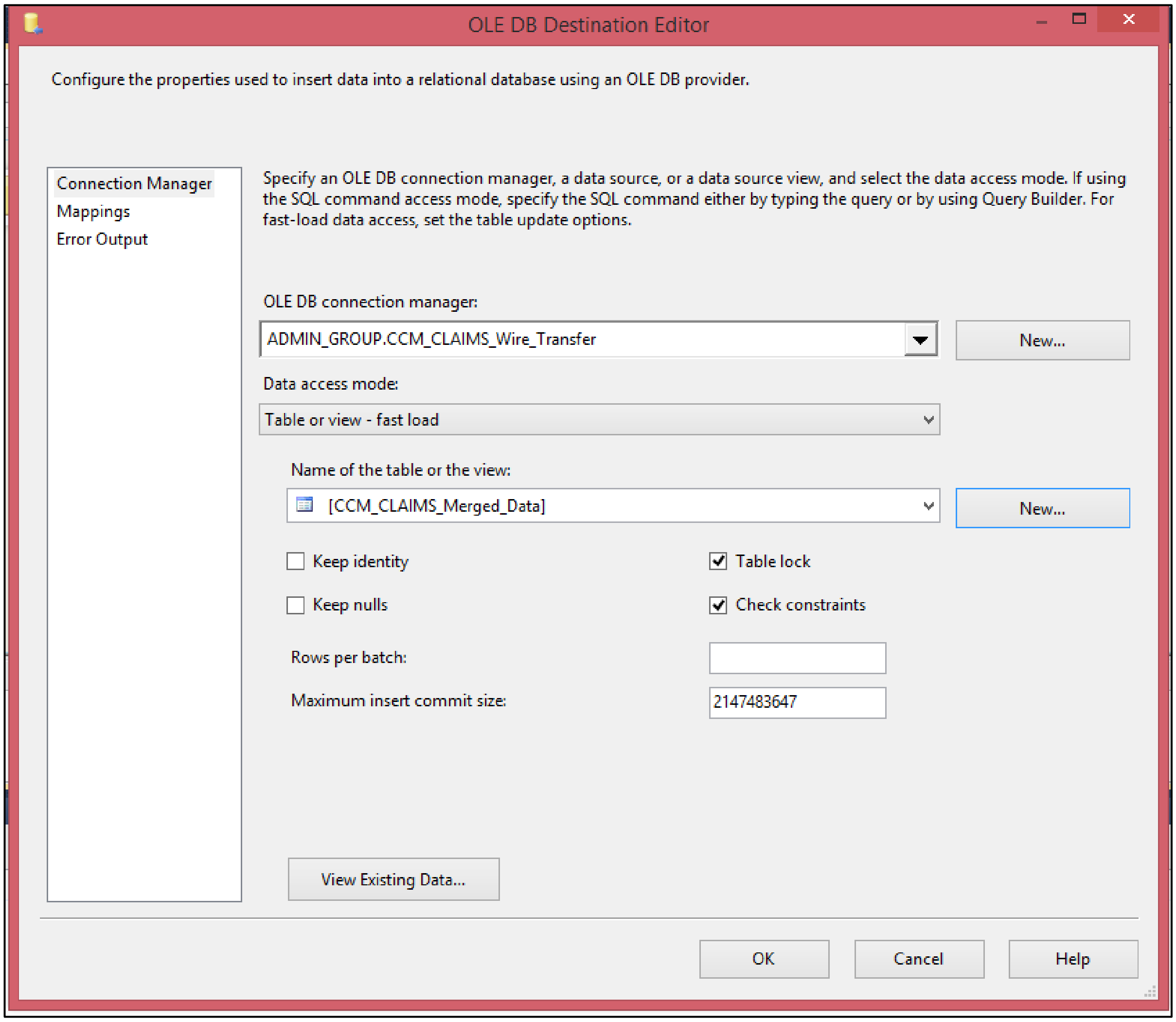
CLAIMS Case_ID

■	Name	Order	Join Ke
□	Case_Id	1	☑
□	Card_Number	0	□
□	Transaction_Amount	0	□
□	Transaction_Date	0	□
☑	Status	0	□

Input	Input Column	Output Alias
CCM Case_ID	Case_ID	Case_ID
CCM Case_ID	Card_Num	Card_Num
CCM Case_ID	validity	validity
CCM Case_ID	Transaction_Amount	Transaction_Amount
CCM Case_ID	Verification	Verification
CCM Case_ID	Transaction_Date	Transaction_Date
CLAIMS Case_ID	Status	Status

Figure 17. Merge Join task configuration.

Step 8*: *Adding and configuring destination. Now the final step in creating the package is moving the merged data to a specified destination. Since the database used in this project is SQL platform, data is moved to the same database but to a different table. A new table is created in OLEDB Destination editor which is shown in the figure below (Figure 18).



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: ADMIN_GROUP.CCM_CLAIMS_Wire_Transfer

Data access mode: Table or view - fast load

Name of the table or the view: [CCM_CLAIMS_Merged_Data]

□ Keep identity ☑ Table lock

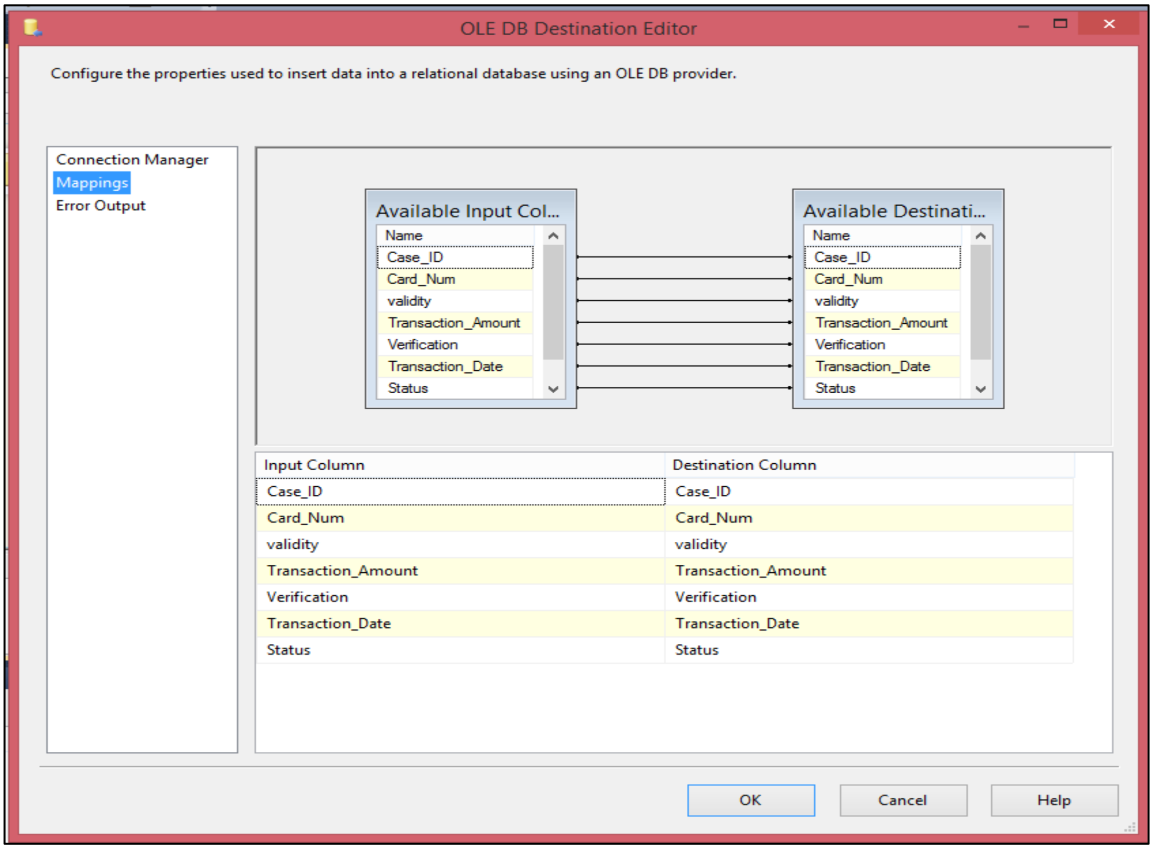
□ Keep nulls ☑ Check constraints

Rows per batch:

Maximum insert commit size: 2147483647

Figure 18. OLEDB destination configuration.

Finally before executing the package mappings are checked to ensure that there are no mismatching of columns as shown in the figure below (Figure 19).



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

Input Column	Destination Column
Case_ID	Case_ID
Card_Num	Card_Num
validity	validity
Transaction_Amount	Transaction_Amount
Verification	Verification
Transaction_Date	Transaction_Date
Status	Status

Figure 19. Mappings for OLEDB destination.

Figure 20 shows how the final package looks like.

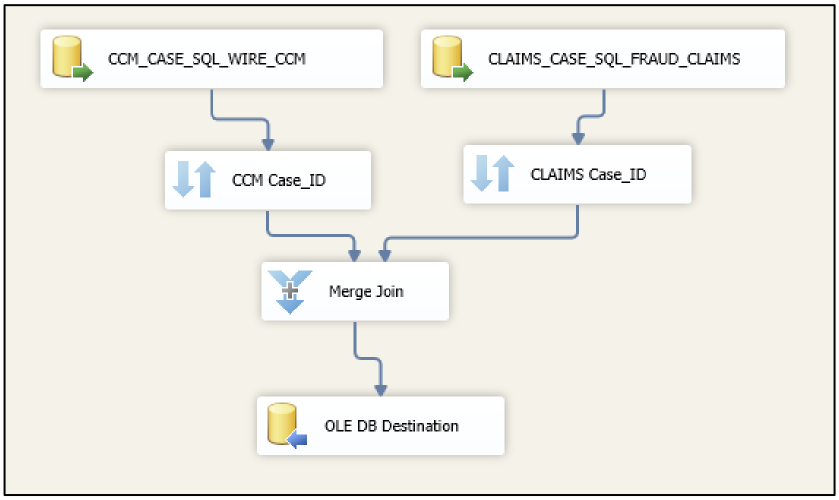
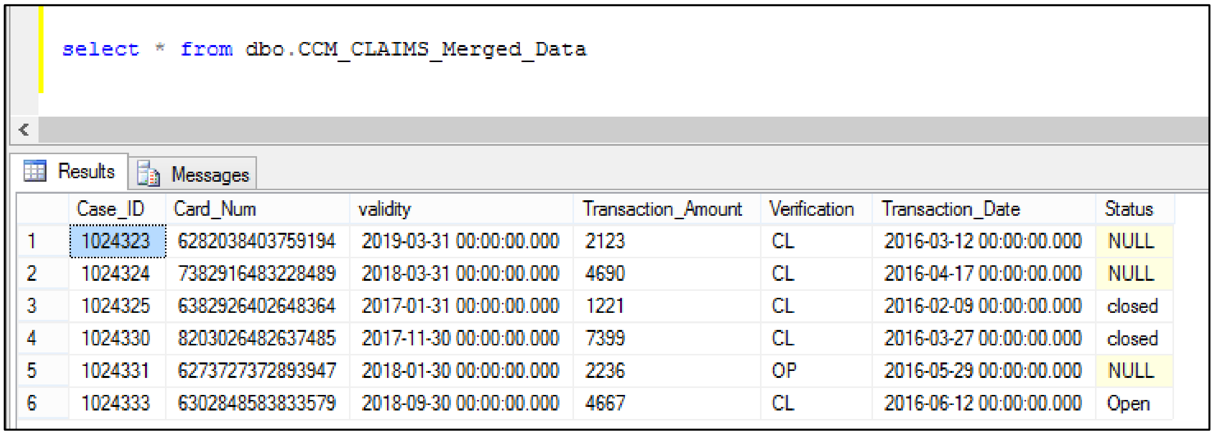


Figure 20. Final package.

Step 9*: *Execute the package. Package is executed by clicking on start debugging from the debug menu or right clicking on the package. Failing to follow any of these steps may have raised an error but since all the steps are followed a new table is created in the database which is specified in the package. Data consistency has been checked by running the package multiple times adding new data.

Data Analysis

Sample data is shown in the figure below (Figure 21) as how the merged data appears in the newly created table in the SQL database.



```
select * from dbo.CCM_CLAIMS_Merged_Data
```

	Case_ID	Card_Num	validity	Transaction_Amount	Verification	Transaction_Date	Status
1	1024323	6282038403759194	2019-03-31 00:00:00.000	2123	CL	2016-03-12 00:00:00.000	NULL
2	1024324	7382916483228489	2018-03-31 00:00:00.000	4690	CL	2016-04-17 00:00:00.000	NULL
3	1024325	6382926402648364	2017-01-31 00:00:00.000	1221	CL	2016-02-09 00:00:00.000	closed
4	1024330	8203026482637485	2017-11-30 00:00:00.000	7399	CL	2016-03-27 00:00:00.000	closed
5	1024331	6273727372893947	2018-01-30 00:00:00.000	2236	OP	2016-05-29 00:00:00.000	NULL
6	1024333	6302848583833579	2018-09-30 00:00:00.000	4667	CL	2016-06-12 00:00:00.000	Open

Figure 21. Merged data table in SQL platform.

Observing the above figure (Figure 21), it is easy for one to analyze the data. Considering the above figure first two cases are reported suspicious transactions but not fraud transactions since their verification shows ‘CL’ which represents closed and they were not entered into CLAIMS database and so the status shows null.

Coming to the next two cases it shows verification as ‘CL’ but the status shows ‘closed’ which means these cases are actually reported as fraud transactions and they have been settled. The next case shows verification as open which means the case has still not been verified by the customer care people whether it is actually a fraud transaction or not and that it why its status still shows null as the case was not opened in CLAIMS. The last case shows it is a fraud transaction since its status was open which means the case has been entered into the CLAIMS database but has not been solved yet.

Summary

This chapter explained in detail how data is integrated using packages, executing package using SQL Server Integration Services and data analysis.

Chapter V: Results, Conclusion, and Recommendations

Introduction

Chapter IV was mainly focused on data presentation and analysis. In this chapter final results, conclusions and recommendations are discussed.

Results

Integrating of CCM data with CLAIMS data was successfully achieved by merging the data from both the applications. This package can be run every day to load the new data and adding it to the current table. It helps in creating a report directly by using the whole table as a dataset and/or exporting the table to an Excel/Word document by querying it. This project was successfully implemented after it was tested by both QA and UAT teams.

Project Questions and Answers.

1. ***What is the efficient way to integrate data?***

Sorting the data before it was merged helped in cutting down the execution time by 77%. Executing the package without sorting the data initially for around 12000 rows took approximately 2.5 minutes whereas execution time for sorted data with the same amount of data took around 35 seconds.

2. ***How is the data consistency achieved?***

Data consistency can be achieved by initially executing the package with small amount of data. Once it is clear that data is correct, server configurations are changed in a way that the package can be run at a certain period of time automatically by scheduling a job so that it adds only new data from the source table avoiding historic data which has already been dumped into the destination table.

3. ***Can the project meet Go Live date?***

Yes, although there are few issues raised at the time testing the project was still able to meet the Go Live date.

Conclusion

Integration of CCM application data with CLAIMS data is a big success which provides the business team with ready access to the filtered data. Not only it provided the business team an easy access to the data but also it saved lot of time and resources to arrange data in an order if it not integrated. Now with just a few clicks data can be viewed in report, word or excel form to make analysis and make proper strategies by the business teams.

Recommendations

Although the project is highly successful it is recommended that the project has 24*7 support for at least 4 months to ensure the data merged is not inconsistent at any point of time.

References

[1] B. Knight, E. Veerman, G. Dickinson, D. Hinson, and D. Herbold, “Professional Microsoft SQL server 2008 integration services,” *Wrox*, First edition, October 2008, pp. 239-247.

[2] MSSQL, “SQL server integration services,” January 16, 2012, https://msdn. microsoft.com/en-us/library/ms141026.aspx.

[3] Microsoft, ”Microsoft SQL server 2012,” September 23, 2013, Typical Uses of Integration Services: http://technet.microsoft.com/enus/library/ ms137795(v=sql.105).aspx.

[4] Experts-Exchange, “A case for ETL tools in database development,” March 5, 2015, http://www.experts-exchange.com/articles/18095/A-Case-for-ETL-Tools-in-Database-Development.html.

[5] Airbrake, “What is Software Development Life Cycle (SLDC)?,” July 9, 2013, https://airbrake.io/blog/insight/what-is-the-software-development-life-cycle.

[6] WebProdigy.com, “What comes first? The chicken or the egg?,” http://webprodigy.com/what-comes-first-the-chicken-or-the-egg/.

[7] Tutorials Point, Agile-Primer, p. no. 2. [8]

Tutorials Point, Agile-Manifesto, p. no. 3.

[9] LINCHPINSEO, “Agile process and method overview,” July 5, 201*2,* http://linchpinseo.com/the-agile-method.