

Data Warehousing & Business Intelligence (IT)

3rd Year, 1st Semester

# Assignment 1

Submitted to

Sri Lanka Institute of Information Technology

IT18125726

Kumarage L.V.A

Weekday Batch

**Step 1: Data set selection**

I have selected IBM Watson Marketing Customer Value Data as data set. It consists of one CSV fie with sufficient data with 24 columns. Furthermore, I have partitioned the main large CSV file into small sub-CSV files. The sub-CSV files consist of new IDs. And, I have manually modified some data records according to the requirements.

The data set was initiated with sufficient data, according to the assignment criteria. It has more the 9000 unique values and it is enriched with transactional data and data hierarchies.

Data Set - <https://www.kaggle.com/pankajjsh06/ibm-watson-marketing-customer-value-data?select=WA_Fn-UseC_-Marketing-Customer-Value-Analysis.csv>

Graphical user interface

Description automatically generated

Chart

Description automatically generated

Graphical user interface, application

Description automatically generated

ER Diagram -

Diagram

Description automatically generated

**Step 2: Preparation of Data Sources**

First, main data set was separated into sub-CSV files (CustomerDetails, InsuaranceDetails, CustomerLocations, PolicyType and Policy) and categorized related data into same csv files. Then csv files were imported into the tables which were in the newly created database called IT18125726\_SourceDB (Except CustomerLocation)

And Customer Location details saved into text file format. This file contains all the customer Location Information.

Table

Description automatically generated

CustomerDetails.CSV

A picture containing text, map, indoor, shelf

Description automatically generated

Insurance.CSV

Table

Description automatically generated Table, Excel

Description automatically generated

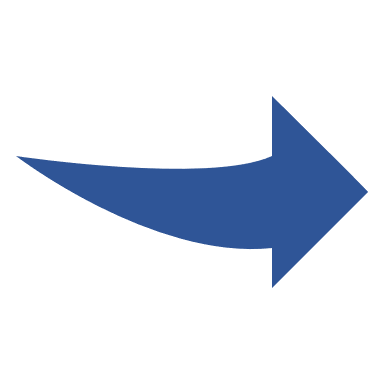
Policy.CSV

PolicyType.CSV

Table

Description automatically generated Table

Description automatically generated with medium confidence



CustomerLocation.CSV

CustomerLocation.txt

Graphical user interface, application

Description automatically generated

IT18125726\_SourceDB

**Database Diagram for IT18125726\_SourceDB:**

Diagram

Description automatically generated

IT18125726\_SourceDB

**Step 03 – Solution Architecture**

**Diagram

Description automatically generated**

Using different processes, architectures, and technologies we can manage data from various sources and convert them to business insights to make decisions, analysis data and report building. This will bring new dimension to the data as well.

**Step 04: Data Warehouse design & development**

**Diagram, schematic

Description automatically generated**

**Assumptions**

* Here I have used snowflake schema for data warehouse design and add 3 dimensions apart from the date dimension.
* I have taken Dim CustomerDetails as slowly changing dimension, customer location code and state can change time to time, and we need to keep track of their historical location details.

Before creating the Insurance fact table & other dimensions, started by creating the Date dimension using ‘DateMaster.sql’ file code dimension using ‘DateMaster.sql’ file code Table

Description automatically generated

Graphical user interface, application, table, Excel

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated Table

Description automatically generated

Table

Description automatically generated

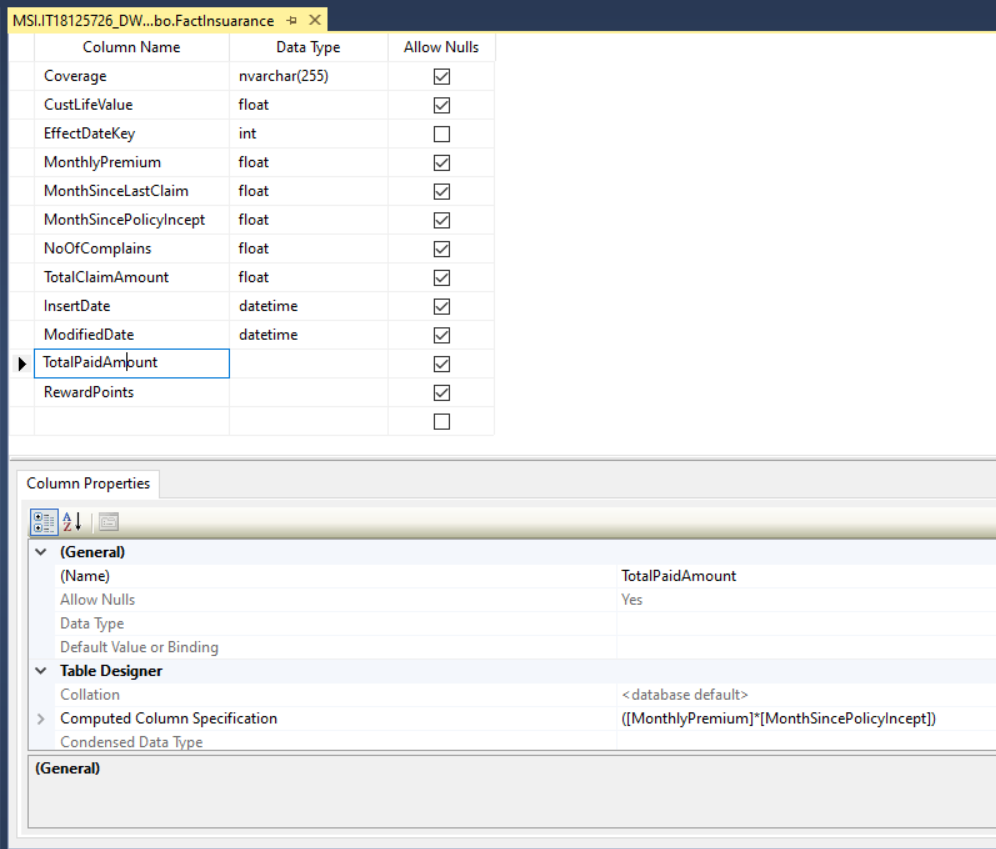
**When crating fact tables I have added two extra attributed which are not include in my data set. I have derived those attributes using existing attributes in data set.**

**These derived attributes are transactional attributes which calculate essential data in fact table.**

Those attributes and their functionality shown below:

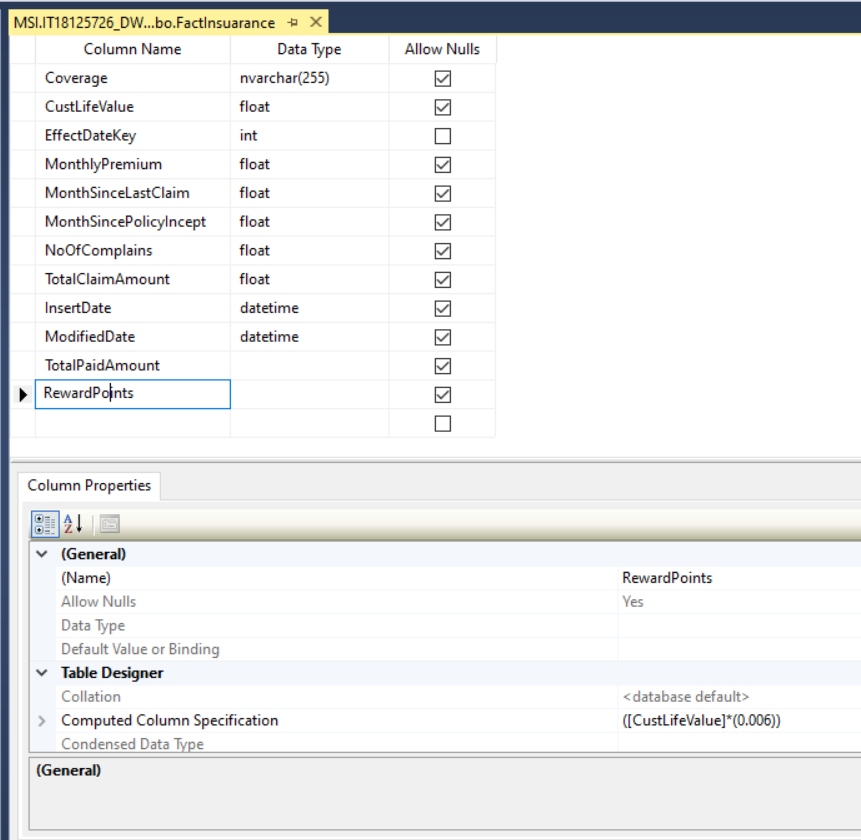
* TotalPaidAmount = (MonthlyPremium \* MonthSincePolicyInception)

This will calculate the total paid amount done by customer, from the day which he or she opened their new coverage.



* RewardPoints = ([CustomerLifeTimeValue] \* [0.006])

This will calculate Reward Points which will be offer to the customer based on their loyalty. I have assumed 0.006 as a constant value which will multiplied by the customer lifetime value.



**Step 06: ETL Development**

**Graphical user interface, application

Description automatically generated**

First extracted all the data from the tables which are in the IT18125726\_SourceDB and CustomerLocation.txt to the separate staging DB called IT18125726\_Load\_Staging as shown in the below using SQL Server Integration Service Software.

Graphical user interface

Description automatically generated with low confidence

IT18125726\_Staging

Extract Customer Details

Graphical user interface, application

Description automatically generated

Used OLEDB data source as dbo.CustomerDetails table in IT18125726\_SourceDB

Graphical user interface, text, application

Description automatically generated

OLE DB destination for create new table as dbo.CustomerStaging in the IT18125726\_Staging database

Graphical user interface, text, application, email

Description automatically generated

Event Handlers

Graphical user interface, text, application

Description automatically generated

Used Execute SQL Task SSIS tool Truncate table for SQL command as truncate table

dbo.CustomerStaging in IT18125726\_Staging Database

Graphical user interface, text, application

Description automatically generated

Customer Address Details Extraction (Data Flow)

Graphical user interface

Description automatically generated

Used Flat file Source SSIS tool, to extract Customer\_addresses.txt data

Graphical user interface, text, application, email

Description automatically generated

Used OLE DB Destination SSIS tool to create new table as StgCustomerAddress to load text file’s data into IT18125726\_Staging Database

In Advanced tab, text file data has columns then every column data type was selected as a String [DT\_STR]

Graphical user interface, application

Description automatically generated

Used Execute SQL Task SSIS tools Truncate table for SQL command as truncate table

Graphical user interface, text, application

Description automatically generated

**Note: Followed exact process to extract other source tables data in to staging**

Data Profiling

Staging table data was used to analyzed and determined what types of transformation was needed to perform on the data.

• Right click on SSIS Packages and New SSIS Package selected.

• In the Control Flow of IT18125726\_Data\_Profiling.dtsx, drag and drop the Data Profiling Task and double click to open the configuration.

• Clicked on Quick Profile button to open Single Table Quick Profile Form.

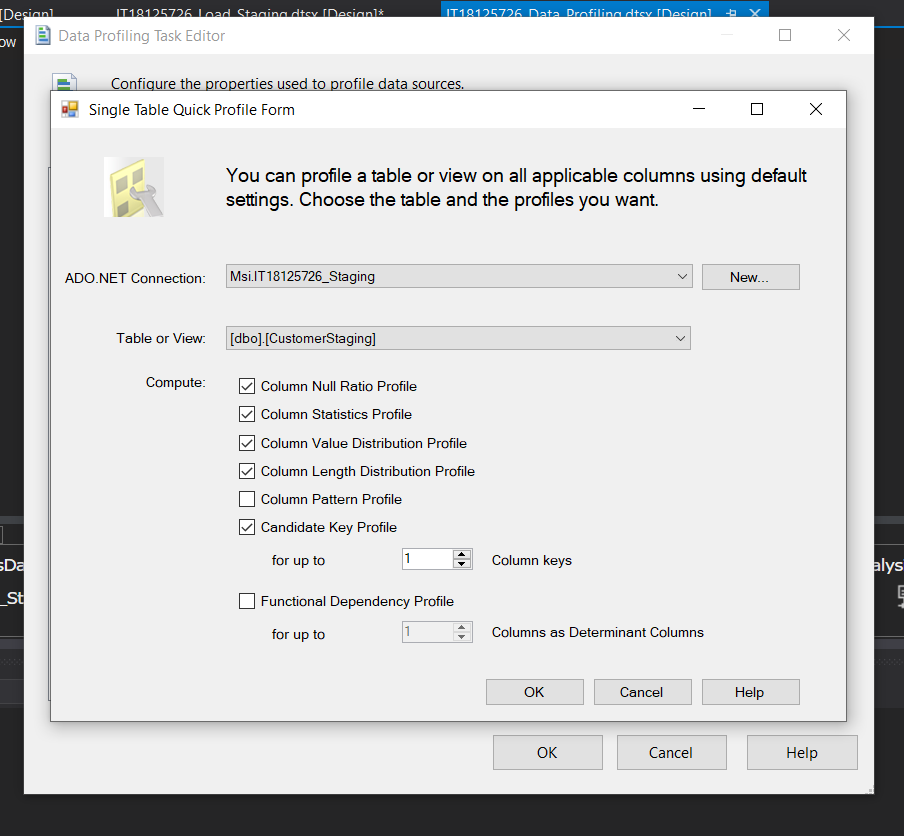
• Clicked on New button and create the connection to IT18125726\_Staging

• From the Table or View dropdown, ConsumerDetails \_staging table selected.

A screenshot of a computer

Description automatically generated

Selected all check boxes and click on OK button to complete the configuration



* Save the package and Run the Data profiling Pack to profile the ConsumerDetails
* Once the green tick appeared, double click the Data Profiling Task and Click on the Open Profile Viewer to view the analyzed data

Graphical user interface, text, application

Description automatically generated

A picture containing text, screenshot, indoor

Description automatically generated

Graphical user interface, text

Description automatically generated

Data Transformation

Customer Data Transformation was created by using below mentioned steps

• Created new package called IT18125726\_Load\_DW.dtsx.

• Then Dragged and dropped a Data Flow Task, renamed it as Transform and Load Customer Data and go the Data Flow tab.

• Dragged and dropped OLE DB Source, renamed as Extract from Customer Staging and configure it to access the CustomerStaging table.

• And I used another OLE DB Source, renamed as Extract from Customer Address Staging and configure it to access the StgCustomerAddress table. And selected all the columns.

• Then I Dragged and dropped two Sort items and connect each OLE DB Source to them.

• After that I Double click Sort that is connected to Extract from Customer Staging and select CustomerID as the Sort option by ticking on the checkbox in from of Customer\_ID Then I did the same for the other Sort item connected to Customer Address Staging.

• Dragged and dropped Merge Join and link above two sort items to the Merge join.

• In the Input Output Selection popup, I have selected Merge Join Left Input.

• After that I dragged and dropped Slowly Changing Dimension item and connect the last Merge Join to that.

• In the SCD Configuration Wizard I set the configurations as below

Graphical user interface, text

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, application

Description automatically generated

Once All Configurations done properly, it will automatically create the Slowly Changing Dimension as shown below.

Graphical user interface, application

Description automatically generated Graphical user interface, diagram, application

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

**Transform & Load Policy Type**

First created a Procedure called UpdateDimPolicyType and executed in the IT18125726\_DW database

CREATE PROCEDURE [dbo]. [UpdateDimPolicyType]

@PolicyTypeID int,

@PolicyType nvarchar (255),

@Response nvarchar (255),

@RenewOfferType nvarchar(255),

@SalesChannel nvarchar(255)

AS

BEGIN

if not exists (select PolicyTypeSK

from dbo.DimPolicyType

where PolicyTypeAlternateID = @PolicyTypeID)

BEGIN

insert into dbo.DimPolicyType

(PolicyTypeAlternateID, PolicyType,RenewResponse,RenewOfferType,SalesChannel,InsertDate,ModifiedDate)

values

(@PolicyTypeID, @Response, @PolicyType, @RenewOfferType, @SalesChannel, GETDATE (), GETDATE())

END;

if exists (select PolicyTypeSK

from dbo.DimPolicyType

where PolicyTypeAlternateID = @PolicyTypeID)

BEGIN

update dbo.DimPolicyType

set PolicyType = @PolicyType,

RenewResponse = @Response,

RenewOfferType= @RenewOfferType,

SalesChannel = @SalesChannel,

ModifiedDate = GETDATE()

where PolicyTypeAlternateID = @PolicyTypeID

END;

END;

Graphical user interface

Description automatically generated

OLE DB Command SSIS tool used to execute, UpdateDimPolicyType procedure, it used to insert data into StgPolicyType to DimPolicyType without data duplication.

Graphical user interface, text, application

Description automatically generated

**Transform and Load Policy**

Used same approached which is followed in loading Policy type data into the data warehouse (using a stored procedure to insert new records and update existing records) , as policy data also does not maintain history

* Using SQL Server Management Studio, I have created the stored Procedure given below, in the IT18125726\_DW database

CREATE PROCEDURE dbo.UpdateDimPolicy

@PolicyID int,

@PolicyTypeID int,

@PolicyName nvarchar(255)

AS

BEGIN

if not exists (select PolicySK

from dbo.DimPolicy

where PolicyAlternateID = @PolicyID)

BEGIN

insert into dbo.DimPolicy

(PolicyAlternateID, PolicyTypeKey, PolicyName,InsertDate, ModifiedDate)

values

(@PolicyID,@PolicyTypeID,@PolicyName, GETDATE(), GETDATE())

END;

if exists (select PolicySK

from dbo.DimPolicy

where PolicyAlternateID = @PolicyID)

BEGIN

update dbo.DimPolicy

set PolicyTypeKey = @PolicyTypeID,

PolicyName = @PolicyName,

ModifiedDate = GETDATE()

where PolicyAlternateID = @PolicyID

END;

END;

Now Drag and drop OLE DB Command task, rename It as ‘Load DimPolicy’ link it with merge join.

After double clicking on Load DimPolicy we can view Advanced Editor for Load DimPolicy window.

Then used bellow code to execute above stored Procedures.

Graphical user interface, text, application

Description automatically generated

Above Stored Procedure ensure no duplicates are entered into the data warehouse table ‘DimPolicy’.

If there is an existing policy record, it will be updated with the latest record coming from staging table ‘StgPolicy’ else, if it is a new record, just insert it.

In Column Mappings tab map, the columns to the variables accordingly. here I had map PolicyTypeSK as the input for @PolicyTypeID.

A picture containing graphical user interface

Description automatically generated

Finally, ‘Transform and Load Policy Data’ data flow design should look like below:

Graphical user interface, diagram

Description automatically generated

**Load to Insurance Fact**

In the IT18125726\_DW.dtsx, add another Data Flow Task and join the previous data flow task with the new data flow task.

Renamed the new Data Flow Task as Transform and Load Insurance data

Then I dragged and dropped the OLE DB source and configure it to fetch data from dbo.StagingInsuarance table

Then I used Sort component to sort data in between sources.

Graphical user interface

Description automatically generated

Effective to Date Lookup –

Here I map Effective to Date Input with Date lookup to obtain date key.

Graphical user interface, application

Description automatically generated

Policy Lookup –

Here I map PolicyID with PolicyAlternateID to obtain PolicySK

Graphical user interface, application

Description automatically generated

Customer Lookup

Here I map Customerid in input columns with CustomerAlternateID in Lookup column to obtain CustomerSK

Graphical user interface, text, application

Description automatically generated

Derived Insert and Modified Date Fields –

Graphical user interface, text, application

Description automatically generated

Take a another OLE DB Destination component and rename it as LooadFactInsuarance

And connected it with above ‘Derive Insert and modified Date Fields’

I mapped the input columns with Destination Columns as below.

Graphical user interface, table

Description automatically generated

Outcome of the Transform and Load FactInsuarance Table:

Graphical user interface, application

Description automatically generatedA screenshot of a computer

Description automatically generated with low confidence

Final IT18125726\_DW.dtsx package Control Flow:

Graphical user interface, text, application

Description automatically generated

At the end of the staging I have connected the data warehousing package to the end of the data staging package using an execute package task editor.

So, when executing the staging it will execute the data warehousing package as well

**References:**

[**https://courseweb.sliit.lk/course/view.php?id=4696**](https://courseweb.sliit.lk/course/view.php?id=4696)

[**https://www.geeksforgeeks.org/etl-process-in-data-warehouse/**](https://www.geeksforgeeks.org/etl-process-in-data-warehouse/)

[**https://www.javatpoint.com/etl-process-in-data-warehouse**](https://www.javatpoint.com/etl-process-in-data-warehouse)

[**https://www.youtube.com/watch?v=oF\_2uDb7DvQ**](https://www.youtube.com/watch?v=oF_2uDb7DvQ)

**END**