# EY-GDS
# D&A SQL DWH Python
# Case Study:02 / 24th December 2023

**Topic Name**

# DW – Building ETL Processes

## TABLE OF CONTENTS

# 1 Introduction

In this case study we will know how to build a Data Warehouse by applying ETL processes. The goal of this case study is to show the usage of the ETL processes in building a Data warehouse specific to business requirements. The ETL tool being used here would be Informatica.

## 1.1 The Project

In the following sections we describe a fictitious project as a case study. We define the scope and objectives and relate them to the requirements of a fictitious client. This is a good test case to see how a Data Warehouse can be built.

## 1.2 The Background

In this fictitious case study, the client has been suffering during many years of not having a computerized system for their day to day processing. With many shops spread across many states, each shop created its own system for their day-to-day activities. It was more and more difficult to follow this method as the accounting was not tracked properly. Hence the company has set out to build a system for tracking the sales data.

## 1.3 The Company

“Sapphire Retail Mart" was established in 2015. For last 5 years, Sapphire Retail Mart is a leading Wholesale & Retail dealer for Consumer goods. Presently we have well established branches running in three major states of US, Texas, Virginia and Pennsylvania.

Furthermore Sapphire Retail Mart is looking forward towards the Exports as well as the International Business for Wholesale, Retail & Export.

# 2 Detailed Requirement

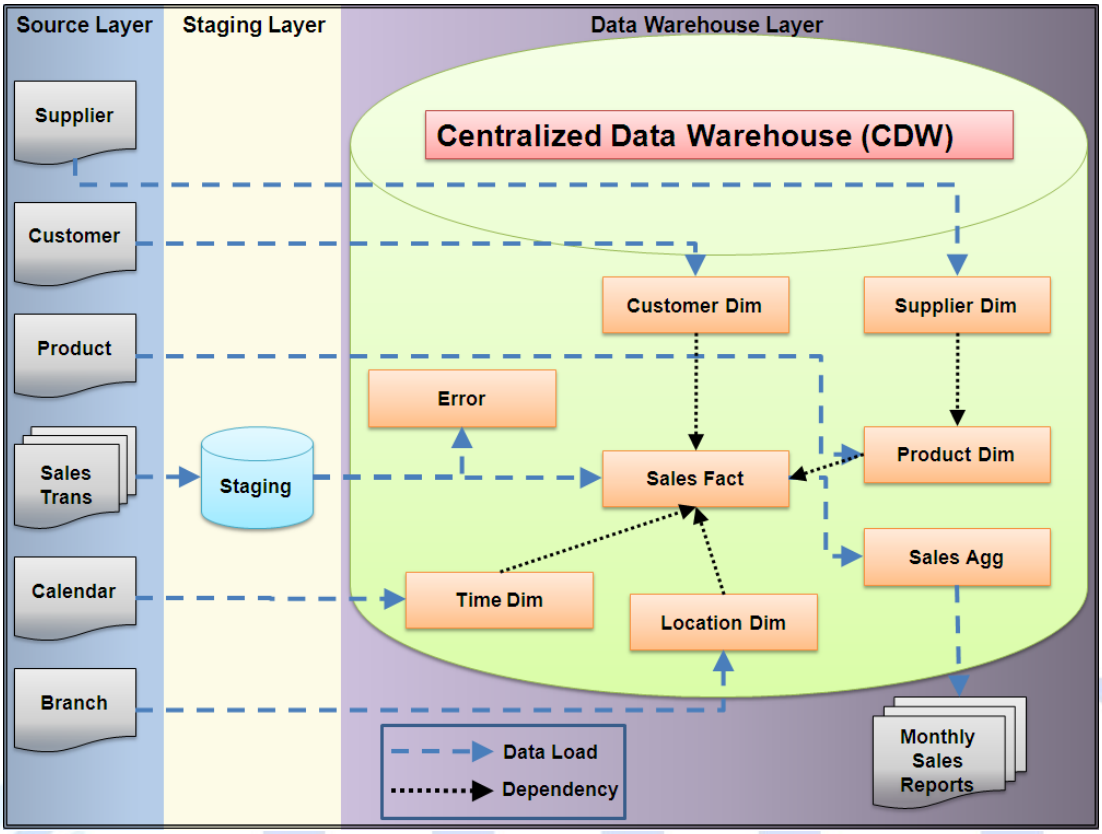


## 2.1 Requirement

Sapphire is planning to build a data warehouse of sales, to enhance their decision support. The business requires a centralized data access for generating various reports to forecast and value their product sales across the branches.

DW Team has won the proposal for the creating ETL Processes to load data into their data warehouse in a daily, monthly, yearly, and on-demand basis.

## 2.2 Application Flow




Sapphire's Data Warehouse is designed using Snowflake schema. They have 5-dimension tables, 1 fact table and 1 aggregate table.

1. The Customer dimension table ***CDW_SAPP_D_CUSTOMER*** is a Type – II SCD table which contains all customers' information. Sapphire receives data from Customer Integration System (CIS). The source file's name is ***CDW_SAPP _CUSTOMER.txt***. The source file is a comma (,) delimited file. The CIS sends the file on a daily basis. History of the customer's needs to be maintained in the table.

2. The Supplier dimension table ***CDW_SAPP_D_SUPPLIER*** is a Type – I SCD table which contains all the suppliers' information for the store. The source file's name is ***CDW_SAPP_SUPPLIER.txt*** and it is pipe (|) delimited. The table is updated/inserted for old/new suppliers respectively on a daily basis and no history is maintained.

3. The Product dimension table ***CDW_SAPP_D_PRODUCT*** is a Type – I SCD which contains all products information. The source file's name is ***CDW_SAPP_PRODUCT.txt*** and it is comma (,) delimited. This table is *dependent on the Supplier table* for the Supplier ID. The supplier table must contain the *respective supplier information before loading product* table. The table is updated/inserted for old/new products respectively on a daily basis and no history is maintained.

4. The Branch dimension table ***CDW_SAPP_D_BRANCH*** is a Type – I SCD which contains all Branch details of Sapphire. *Data is added to this table, only when a new branch is being opened by Sapphire*. So, the load schedule would be *on demand*. The source file's name is ***CDW_SAPP_BRANCH.txt*** and it is pipe (|) delimited. The table is updated/ inserted for old/ new branches respectively and no history is maintained.

5. The Time dimension table ***CDW_SAPP_D_TIME*** is a static table which contains the time key and other relevant information. This table is loaded once in a year from the source file

***SRC_CALENDER_FILE.txt*** which is fixed width file.

6. The Sales staging table ***CDW_SAPP_STG_SALES*** receives the daily transaction data from the different branches as transaction files in different formats (Two delimited & one fixed width) ***CDW_SAPP_F_SALES_BR_XX.txt*** and the delimiter used is comma. The information from different files is standardized in the staging table. This table is refreshed every day. The data is loaded with all related dimension information from the respective dimension tables in the warehouse. Records containing invalid information will be sent to the ***CDW_SAPP_ERR*** error table. The two delimited files are joined before loading into the staging area.

7. The Sales fact table ***CDW_SAPP_F_SALES*** contains all the transactions loaded from the staging table ***CDW_SAPP_STG_SALES*****.** This table is loaded o*nly after the staging table has been loaded completely from the transaction files arriving from all the different branches.*

8. The Sales Aggregate table ***CDW_SAPP_F_AGG_DATA*** pulls data from the Sales fact table ***CDW_SAPP_F_SALES***. The table is aggregated by branch_code & product_code for the current month. Sum of sales price of each product in each branch & the total quantity of each product in each branch is stored in the table.

9. A monthly report on sales, most profitable branch & most selling product is generated.

## 2.3 Table Definitions & Mappings

| Table Name | Table Type | Table Description |
|---|---|---|
| CDW_SAPP_D_CUSTOMER | Type 2 Dimension | Contains Customer data with history |
| CDW_SAPP_D_SUPPLIER | Type 1 Dimension | Contains Supplier data |
| CDW_SAPP_D_PRODUCT | Type 1 Dimension | Contains Product data |
| CDW_SAPP_D_BRANCH | Type 1 Dimension | Contains Branch data |
| CDW_SAPP_D_TIME | Static Dimension | Contains Time data |
| CDW_SAPP_STG_SALES | Staging | Staging Area for the Sales Fact table |
| CDW_SAPP_F_SALES | Fact | Contains Sales Transactions happened across all branches, with history |
| CDW_SAPP_F_AGG_DATA | Aggregate | Contains monthly aggregates from Sales data |
| CDW_SAPP_ERR | Error | Contains Error data from Sales Staging Load |

The attached spreadsheet contains the table definitions and the mapping logics in details.



## 2.4 Deliverables

1. Create a technical specification document and an understanding document.
2. Create Unit test case document before starting the development.
3. Mappings and workflows are to be developed as follows:
    a. Customer Dimension table has to be loaded daily as per the mapping document.
    b. Supplier Dimension table has to be loaded daily and it should be loaded before the product dimension table.
    c. Product Dimension table is a daily load table which has a dependency on supplier table.
    d. Branch Dimension table should be loaded on-demand.
    e. Time Dimension table has to be loaded on a yearly basis, at the beginning of every year.

f. For CDW_SAPP_STG_SALES table load, create a single mapping and three sessions to accommodate three source files of different formats. The above-mentioned dimension tables should be loaded before the SALES fact table load. Use the ERROR_TABLE in the staging load as per the mapping document.

g. CDW_SAPP_F_SALES has to be loaded from the staging table after all the sales data has landed in the staging zone.

h. CDW_SAPP_F_AGG_DATA is a monthly load table which will hold summary information for every month.

i. Two downstream files (MNTH_SALES_RPT_BRANCH_FILE & MNTH_SALES_RPT_PRODUCT_FILE) are generated which contain the branch which yields highest revenue and the top five selling products respectively.

4. Create a integration test case document. (Workflow or system as a whole).
5. Create an implementation document which has the saved the session logs and succeeded screen shots of workflow monitor, to ensure the completion of development.

# 3 Case Study Team Contacts

## 3.1 Standard Owners

## 3.2 Standard Owner Delegates

# 4 Document Changes

| Date | Name | Reviewer Name | Changes |
|---|---|---|---|
| | | | |

# 5 Appendix

## 5.1 DDL Scripts

## 5.2 Source File Structure