BI Support Case Study

# Setup

The provided zip archive contains a sample SQL Server 2019 database backup (CaseStudy.bak), and a folder (import) with CSV files. Extract and mount the backup file to a SQL Server installation and complete the exercises included in this case study.

Complete exercise one and two for consideration. The challenge and exercise three are optional.

***Note: SQL Server 2019 Developer and Visual Studio Community 2019 are free to download.***

To improve case study portability, the provided database is divided into an EDW and a staging schema representing the staging area and EDW database.

For simplicity, the Payment Type and Dining Option dimensions presented in this case study contain only one attribute and a surrogate key. These should not be considered degenerate dimensions. Assume that the single attribute is the natural key, and a separate process manages the dimension load. The fact load should properly handle late-arriving dimensions as needed.

The output of this case study should be three SQL scripts that when run against a copy of the original case study database produce the desired outcome for each exercise along with an SSIS package (for the second exercise).

# Background

The business has provided your team with the following definitions:

* *Unit*: a single location for a brand
* Check: A single transaction from the POS
* *Check Number*: the sequential order of checks for a given unit and day
* *Check Time*: the number of minutes from midnight that a transaction occurred

Additionally, the data architect has provided the following details about the EDW:

* *Unit Dimension*: Type II using StartDate, EndDate, and Active columns to track history
* *Date Dimension*: Type 0 including a 4-4-5 fiscal calendar hierarchy
* *Payment Type and Dining Option Dimensions*: Type I

Your manager has requested the following:

* New code introduced to the SQL environment should follow the existing style, format, and documentation pattern as existing code as much as possible
* All changes to production should be provided via deployment script (SQL) and SSIS packages.

Assume that an external process truncates the staging tables after each successful load of the EDW and that the same process repopulates the last *14* days of data to staging on each subsequent load. This strategy was chosen after observing late-arriving sales data and changes to sales data up to *14* days after the first load of a check from the POS to staging.

# Exercise One

A SQL Server job associated with loading edw.FactCheck began failing on 10/14 after a developer released a change to production. The developer is unavailable, and you are tasked with troubleshooting the job. It has been several weeks since new data has made it to the fact table. Since then, new data has been accumulating in staging. You’ve isolated the error to the step that calls the stored procedure edw.spLoadFactCheck.

* Identify and resolve the issue
* Your solution should correct the underlying issue and load any missing data to the fact table
* Save your solution as a single SQL script “Exercise One.sql”

# Exercise Two

The executive leadership team wants to provide unit operators with estimated daily revenue goals. They have provided several CSV files with the annual per-unit revenue goal by brand and year. The data architect has created a fact table to hold the information and asked you to create an SSIS package and stored procedure to load those CSV files to a staging table and transform them into the fact table edw.FactRevenueGoal.

* Create a table in the Stage schema to hold the data from the CSV files
* Create an SSIS package to load the table from the CSV files
* Create a stored procedure to load the fact table from the newly created staging table
* Consider the grain of the fact table: Daily Revenue Goal by Unit
* Consider the revenue goals in the CSV files represent fiscal year and not calendar year
* Consider the revenue goals in the CSV files represent the fiscal year goal for a single unit

Your colleagues noticed that the date key column in the new fact table is improperly named “CalendarKey” instead of “DateKey”. Update the column name in the table as part of your deployment script.

## Challenge - *Optional*

The executive leadership team reviewed the results of the fact table load and noticed that daily revenue goals are showing for units prior to their opening date. Your manager has asked that you adjust your fact load such that daily revenue goals only get inserted for units beginning on the unit open date. For this exercise, you do not need to consider the unit closing dates.

At the end of this challenge, the daily revenue goals for all units of the same brand and for the same year should match, but the fiscal year total goals for each unit will vary depending on how many days that unit was open during that fiscal year.

# Exercise Three - *Optional*

Several new brands have been acquired and those brands use a different POS (Aloha) than the existing brands. Another BI developer has configured a process to load that data into the staging area as the table stage.CheckAloha. Historic data has been populated into the table and is accumulating in that table until a process is built to integrate it into edw.FactCheck. You have been tasked with this integration. Modify the existing stored procedure to include this new data in the fact load. While implementing your solution, consider the following:

* Aloha does not have a check number, but check time is guaranteed unique for a unit and day
* The check number should represent the check number for the unit and for the day and should be a sequential whole number (i.e. 1, 2, 3, … 10)
* Aloha has introduced new dining options and payment types that should be added as part of this load and on an ongoing basis
* Aloha sales data is collapsed to a single caret and pipe delimited string per unit per day
* Each data element within this string is prefixed by its name and a colon and data elements are comma delimited