Deloitte.

Detailed Order Recognition (DOR) concept and methodology

Author:
Durgesh Girdhar Hedau – dgirdharhedau@deloitte.com

Sponsor:
Robert T Willis - robwillis@deloitte.com
Gaurav Tyagi – gtyagi@deloitte.com
Rajavardhana K Reddy - rkarna@deloitte.com

Contents

1. Project Background

Client is multinational enterprise information technology company that provides products and services geared toward the data center such as servers, enterprise storage, networking, and enterprise software. Purpose of the project was to accurately recognize the Custom SOW (Statement of work) Orders, Service contracts and Service Orders for forecasting and linkage to sales crediting. The model is the classic example for near to real time reporting for the recognized Orders or Service contracts. Thereby, supporting business users from Finance and Services to take their decision on performing the respective billing on a day-by-day basis.

2. Problem Statement

The order capturing and submission process and how DOR model recognizes and resolves various incoming Orders/Service Contracts from S4 HANA system for specific order types in EAP(HIVE) platform for recognizing the revenue at specific point of time. The amount of revenue arising on a transaction is usually determined by agreement between the parties involved in the transaction. When uncertainties exist regarding the determination of the amount, or it's associated costs, these uncertainties may influence the timing of revenue. Process goes with meeting definite predefined criteria ultimately contributing to timely revenue recognition.

3. Key Terms

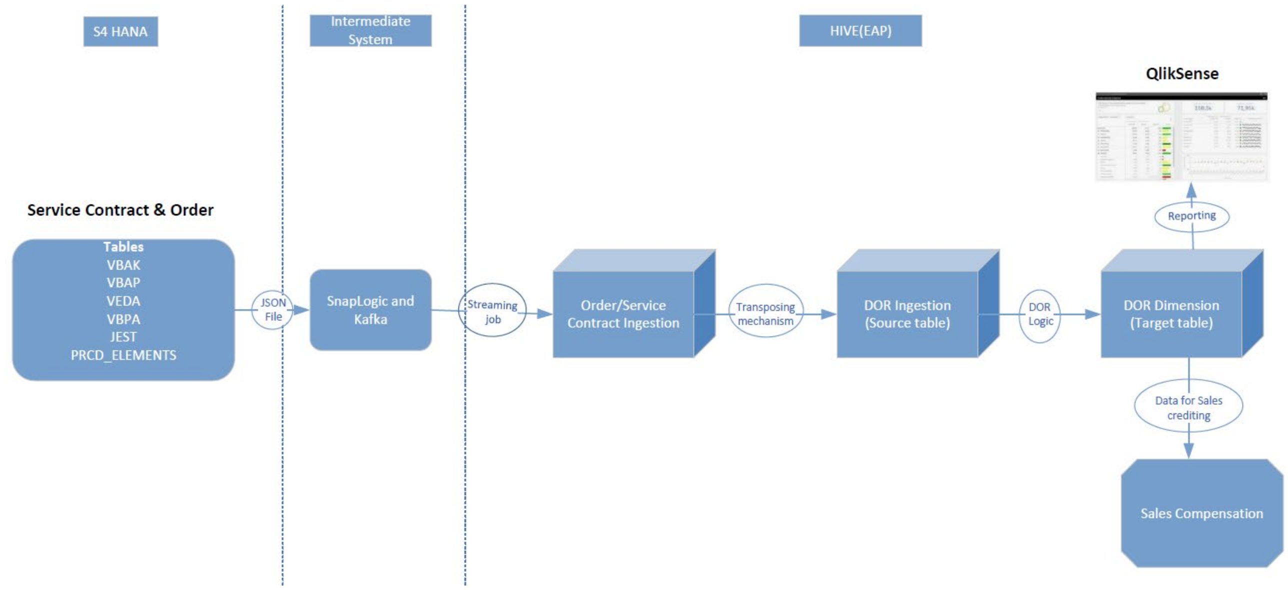
1. CDS – Core Data Services are virtual data models of SAP HANA which allows direct access to underlying tables of the HANA database. SAP CDS Views aim to push logic from the application server to the client-side and database.
2. NACE Configuration – To link the Application type, Output type and Processing routine (driver program) to generate the Output.
3. Kafka topic - Open-source Stream Processing and Management Platform that receives, stores, organizes, and distributes data across different end-users or applications.
4. HIVE – HIVE is a data warehouse system which is used to analyze structured data. It is built on top of Hadoop.
5. EORD- Earliest order recognition date specifies the date by when the Order should get recognized. EORD is logically derived based on the window determination (as per header start and end date) and the line-item duration for the Order/Service Contract.
6. OCM- Order criteria met represents the flag which signifies if all the criteria are met to recognize the Order/Service Contract.
7. ORF- Order reportable flag is an indicator which signifies whether the Order/Service Contract can be reportable or linked for Sales crediting when recognized.
8. User statuses-
   a) CACT(Active)- Header level status which defines the Order/Service Contract is active and can be considered for DOR processing,
   b) INAC(Inactive)- Header level status which defines the Order/Service Contract is inactive and should not be considered for DOR processing,
   c) GT Block- Header level status for representing Global Tread Block which will restrict Order/Service Contract to consider for DOR processing.
   d) Credit Block- Header level status for representing credit block on specific country or with credit amount going beyond the predefined limit,
   e) DOR relevant- Item level status for defining the Line-item is DOR relevant,
   f) NDOR- Item level status for defining the Line-item is non-DOR relevant
9. Completion/Incompletion- Defines the completeness or incompleteness of the Order/Service Contracts and one of the conditions for OCM to be 'Y'.

10. Prepaid flag- Indicates the payments are made in advance for the Order/Service Contract and accordingly considered for the recognition process.
11. Change Order- Represents the change for the Order/Service Contract which provides flexibility to Business to perform changes on payment terms.

4. Pre-Requisites

1. SnapLogic system (intermediate system) to support ingestion of streaming files from S4 to HIVE(EAP) to provide near to real time reporting.
2. HIVE ingestion architecture to flatten the file received from S4 in two-dimension(tabular) view in HIVE(EAP).
3. HIVE transposing mechanism to segregate the required fields from the single attribute from the source to perform the necessary logic.
4. HIVE source and target tables to compare the latest record with previous record to perform necessary steps from the DOR logic.
5. Reporting tool like Qlik sense, Power BI, etc.
6. Minimum System requirement:
   S4/HANA: SAP 7.3 SP 10
   SnapLogic: Snaplex -13393 - 4.30 Patch 1 (Deprecated)
   HIVE: NAME - CentOS Linux; Version - 7 (Core); spark version - 2.3.0.2.6.5.0-292; Beeline version - 1.2.1000.2.6.5.0-292 and Hadoop (Hortonworks) - 2.7.3.2.6.5.0-292
   QlikSense - QS Sept 2020 patch 4
7. Workflows/jobs set up to ensure timely refresh of the records on the respective layers with appropriate queue/memory allocation.

5. Architecture



Process in S4

1. Maintain output parameter in t-code VV11 for Orders/Service Contracts country codes with respective document type.

2. NACE configuration for Application V1 is required to call the program whenever the respective output is triggered via t-codes VA01 & VA02 (for Orders) and VA41 & VA42 (for Service Contracts) and generate the JSON file that get pulled from Intermediate system (SnapLogic).

Process in Intermediate system (SnapLogic)

3. Pull the JSON files from S4 by maintaining the persistent connectivity.
4. Maintain the records in Kafka topic with having minimum retention period of 7 days.

Process in HIVE(EAP)

5. Ingest JSON files from Kafka topic and perform flattening of records by updating it into two-dimensional tables for further processing.
6. Transposing or bifurcating the fields as required for DOR logic.
7. Performing the DOR logic and checks for the recognition of Order/Service Contract.

Process in QlikSense

8. Maintaining the reports as per the Business requirement which should at least refresh every 2 hourly.

Objects created in S4, SnapLogic and HIVE(EAP) systems:

System	Object type
S4	Standard ABAP program called via NACE configuration
SnapLogic	Kafka topic
HIVE(EAP)	Ingestion tables (History, Error, Raw, Refined tables)
HIVE(EAP)	Temp (staging) tables for transposing fields and for performance optimization
HIVE(EAP)	DOR Dimension (Reporting) table
QlikSense	DOR Log file, Missing DOR, Header level and Item level reports

9. Available Options

1. **Creation of Custom CDS View in S4** – Custom CDS Views can be created to provide the report by logically joining required tables with having complex DOR derivations in S4.

Pros:

- Saves time by performing the joins and derivation in S4 database.

- Near to real time reporting.

Cons:

- Complex logic slows up the performance while generating the report.
- Large amount of data will be pulled due to 300 fields in CDS (~50 million records).
- Difficult to achieve the consolidated view with having 300 fields.
- Multiple CDS views will be required for different reports.
- No feature of debugging.

2. **Creation of Custom CDS View in S4 and then consuming to HIVE(EAP)** –

Custom CDS Views can be created by logically joining required tables in S4. These CDS views are then exposed to HIVE(EAP) through Data Integration (DI) method. Ingested views are combined to have all the required fields available for performing the DOR logic.

Pros:

- HIVE capability can be used for analysis.
- HIVE support huge data volume processing with capacity of handling ~1200 attributes and provides optimized processing of the records through various functionality of HIVE.
- Spark SQL is used for optimized execution of the complex logic.
- Automation of data through scheduling the workflows/jobs as per the Business requirement.
- Layered structure provides easy debugging of issues with an additional feature of error handling.
- Housekeeping activities can be performed according to the business requirement by performing conditioning and filtering on the records.
- Support execution of complex logics without having any impact to the performance.

Cons:

- Synchronizing and combining the multiple ingestions will add up to processing time.
- Missing records in case of unexpected technical errors while generating or delay in generating one or other files out of multiple CDS views from S4 will eventually results in incomplete reporting.
- Data availability is in near real time.

3. **Creation of consolidated JSON file through NACE configuration and then consuming to HIVE(EAP)** –

NACE configuration for Application V1 is called through ABAP program whenever the output is triggered via t-code VA01 & VA02 (for Orders) and VA41 & VA42 (for Service Contracts) and generate the consolidated JSON file that get pulled from Intermediate system (SnapLogic). JSON file is then pulled from respective Kafka topic from Snaplogic in HIVE(EAP) for performing the DOR logic.

Amongst the available options, option 3 was chosen for implementation due to below reasons:

Pros:

- HIVE capability can be used for analysis.

- HIVE support huge data volume processing with capacity of handling ~1200 attributes and provides optimized processing of the records through various functionality of HIVE.
- Spark SQL is used for optimized execution of the complex logic.
- Parallelism mechanism is used while running the Spark job which make sure every partition task gets single core for processing.
- Optimized Row Columnar (ORC) file is used which improves the performance while reading, writing, and processing the data.
- Intermediate temp tables (staging tables) are created to store the data temporarily while performing the complex DOR logic sequentially which distributes the load on memory by avoiding long running jobs or job failure post long run.
- Automation of data through scheduling the workflows/jobs as per the Business requirement.
- Layered structure provides easy debugging of issues with an additional feature of error handling.
- Housekeeping activities can be performed according to the business requirement by performing conditioning and filtering on the records.
- Support execution of complex logics without having any impact to the performance.

Cons:

- Data availability is in near real time.

7.Solution Overview

ABAP driver program is written in S4 system which calls NACE configuration for Application type 'V1' (for Sales) when the output is triggered with an immediate trigger mode from VA01 & VA02 (for Orders) and VA41 & VA42 (for Service Contracts), which generates the consolidated JSON file per Order or Service Contracts.

Intermediate system (SnapLogic) pulls the JSON file and stores the data in Kafka topic with an archival period of 7 days. With specific configuration maintained in EAP, the data gets pulled/ingested in HIVE(EAP) system with the streaming jobs. The streaming job makes sure to pull the data in HIVE(EAP) as soon as data is available.

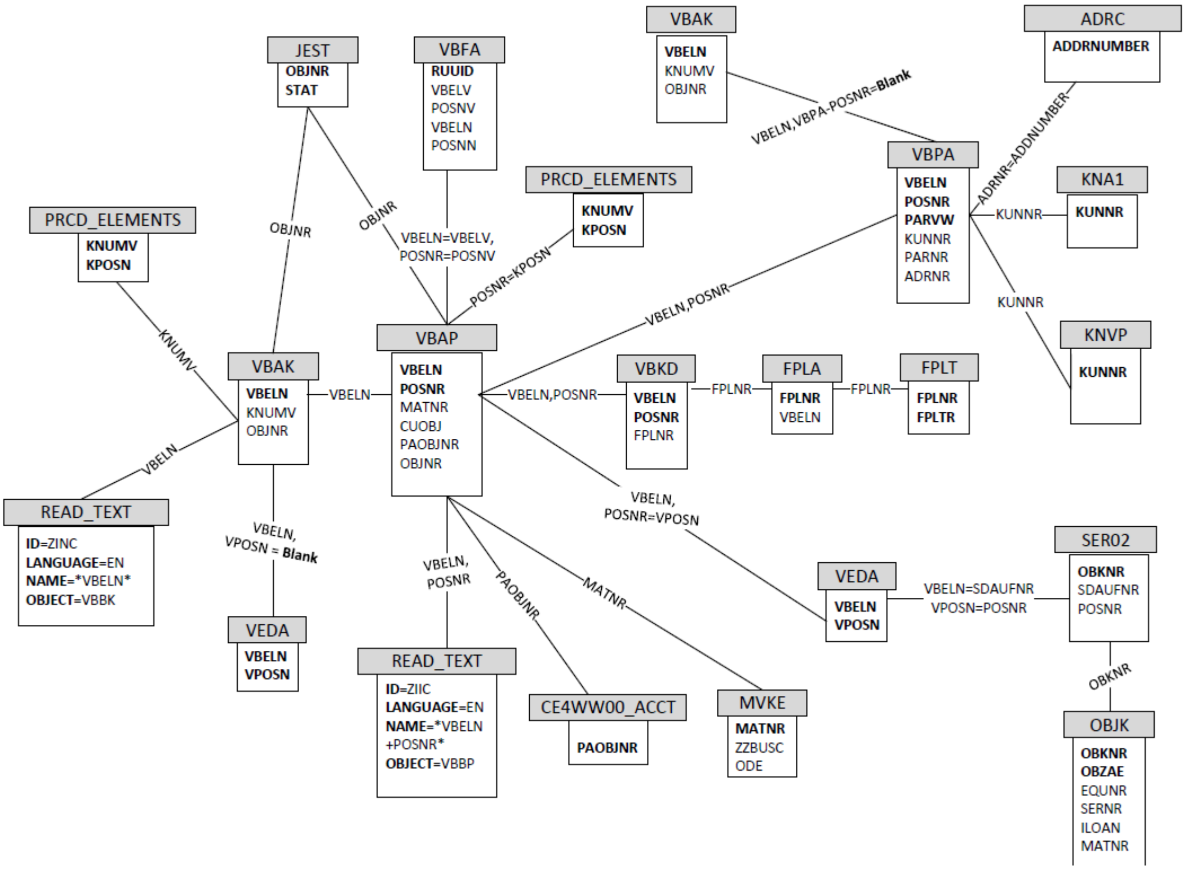
Ingested JSON file is then flattened in two-dimensional table with specific architecture consists of Error, Raw and Refined tables. As S4 tables such as VBPA for Party function and JEST for User status contains the relevant information for various partner functions and statuses respectively in a single attribute, logical bifurcation is required through Transposing mechanism to get the individual fields available to perform necessary derivations.

Transposed fields are then made available to the DOR consumption layer by creating the DOR specific Refined table. While writing the code this Refined table is also termed as Source table for easy comparison with already existing records in target table. Target table is DOR specific Dimension table for reporting which is structured in such a manner to keep all the transactions. DOR main logic including the currency conversion is written between source (DOR refined) and target table (final reporting table) with having intermediate temporary (staging) tables created to distribute the load on memory/queue while processing the logic sequentially before making it available in target table. Temporary table stores the data on go basis while getting the incremental/delta records and deletes it with execution of the next run.

QlikSense capability is used to report the data by fetching the records from DOR dimension table from HIVE(EAP). Various sheets to maintain all DOR transactions, Header and Line-item data, missing or error records are created according to Business need.

7.1 Process Steps

1. Driver ABAP program combines the data logically when NACE configuration for Application type 'V1' is called into a single JSON per Order/Service Contract when triggered in immediate mode. Below is the joining condition with details between the various tables.



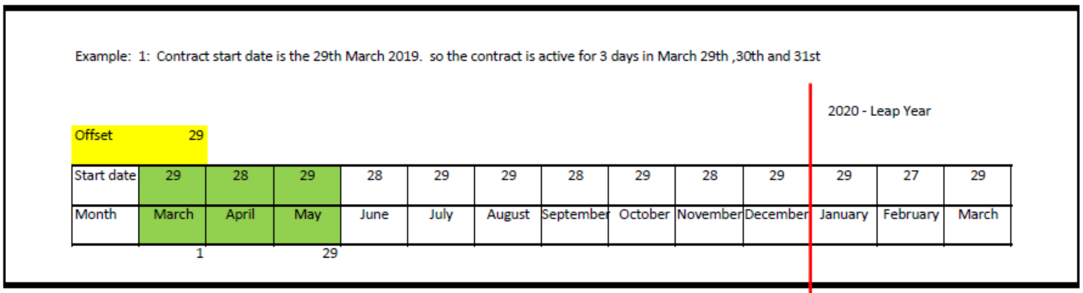
2. EAP then ingest the JSON file from SnapLogic to flatten it further into two-dimensional Refined layer.
3. Data from VBPA and JEST table are transposed/bifurcated into respective User statuses and Partner functions and loaded into DOR Refined (source table for DOR calculations) with rest of the ~180 relevant fields.
4. DOR Dimension (target table for DOR calculations) is then refreshed from DOR Refined post execution of DOR logic in sequential manner by loading intermediate temp/staging tables (helps in reducing the memory consumption by executing the piece of code at a time on the go basis) as per the below attached flow which is then available for reporting.

   Please check and download the "DOR logic and flow diagram" from the title page for below attached details.

DOR logic.pdf

5. Window determination is an important factor while performing the DOR logic which is determined based on Order/Service Contracts header start and end date. Window is of maximum 12months, and the current window is considered for performing the DOR logic and the future windows will wait until the EORD becomes equals to or less than current date.
6. Offset determination of the start of subsequent months when the last day of the initial month is greater than the 28th - which the last three active days is explained in attached excel below.

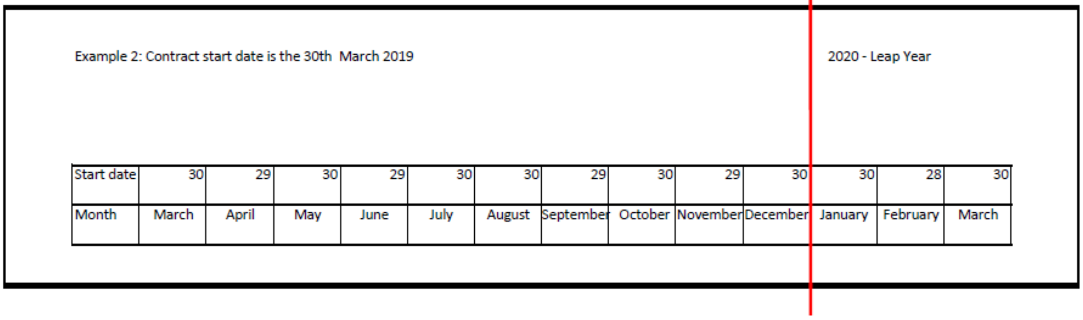
Please check below for the details with various scenarios.



Example: 1: Contract start date is the 29th March 2019. so the contract is active for 3 days in March 29th ,30th and 31st

2020 - Leap Year

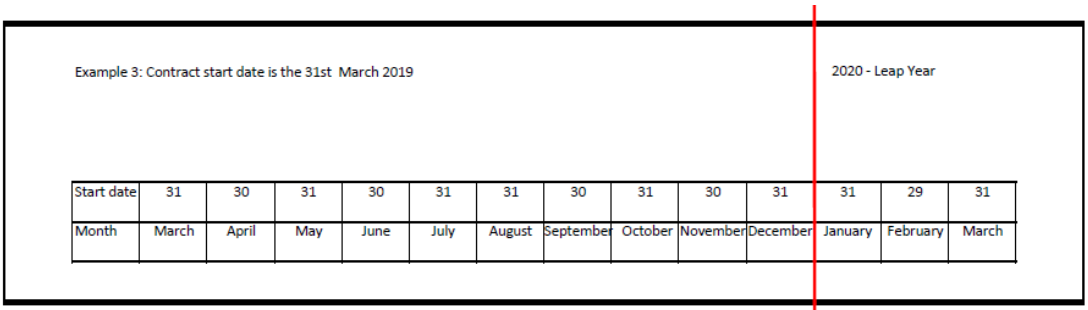
Offset	29											
Start date	29	28	29	28	29	29	28	29	28	29	29	27	29
Month	March	April	May	June	July	August	September	October	November	December	January	February	March
	1		29										



Example 2: Contract start date is the 30th March 2019

2020 - Leap Year

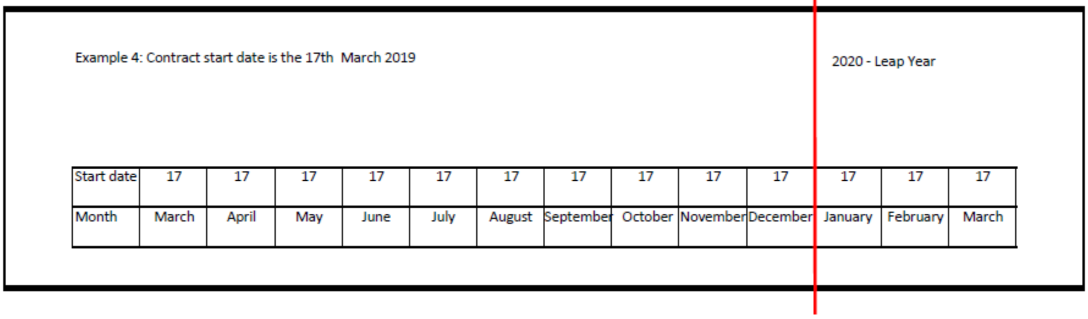
Start date	30	29	30	29	30	30	29	30	29	30	30	28	30
Month	March	April	May	June	July	August	September	October	November	December	January	February	March



Example 3: Contract start date is the 31st March 2019

2020 - Leap Year

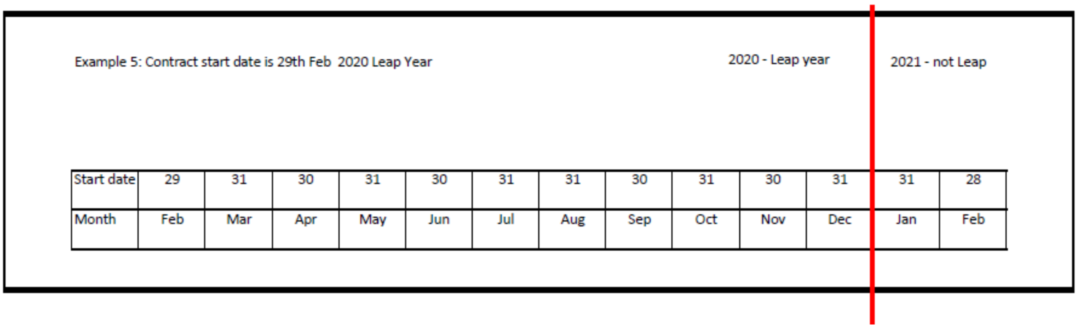
Start date	31	30	31	30	31	31	30	31	30	31	31	29	31
Month	March	April	May	June	July	August	September	October	November	December	January	February	March



Example 4: Contract start date is the 17th March 2019

2020 - Leap Year

Start date	17	17	17	17	17	17	17	17	17	17	17	17	17
Month	March	April	May	June	July	August	September	October	November	December	January	February	March



Example 5: Contract start date is 29th Feb 2020 Leap Year

2020 - Leap year

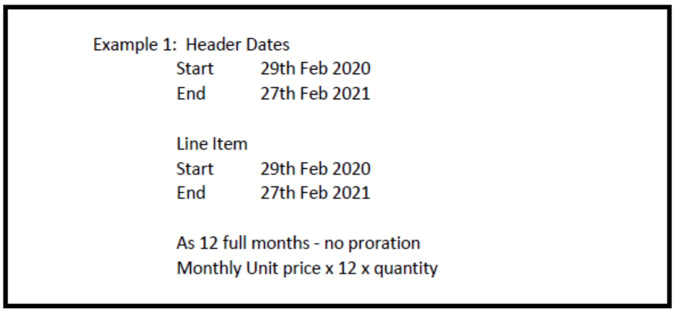
2021 - not Leap

Start date	29	31	30	31	30	31	31	30	31	30	31	31	28
Month	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb

7. Various scenarios of proration calculation when Header start and end dates doesn't match with Line-item start and end dates is explained in attached excel below.

Please check below for the details with various scenarios.

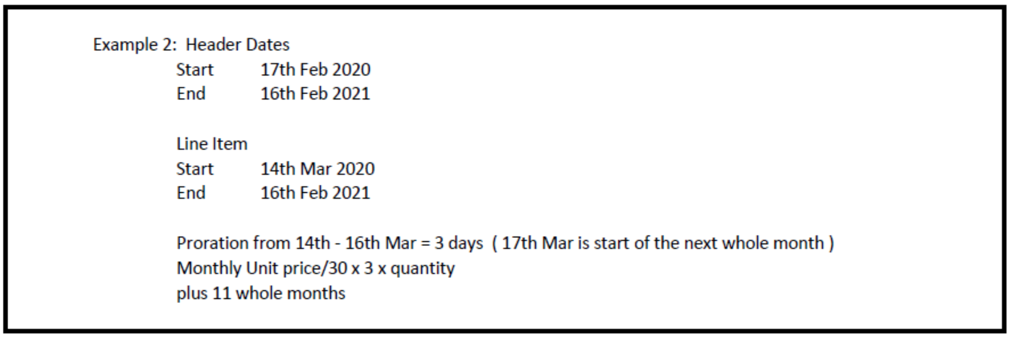
Proration examples:



Example 1: Header Dates
Start 29th Feb 2020
End 27th Feb 2021

Line Item
Start 29th Feb 2020
End 27th Feb 2021

As 12 full months - no proration
Monthly Unit price x 12 x quantity



Example 2: Header Dates
Start 17th Feb 2020
End 16th Feb 2021

Line Item
Start 14th Mar 2020
End 16th Feb 2021

Proration from 14th - 16th Mar = 3 days (17th Mar is start of the next whole month)
Monthly Unit price/30 x 3 x quantity
plus 11 whole months

Example 3: Header Dates
Start 17th Feb 2020
End 16th Feb 2021

Line Item
Start 17th Feb 2020
End 24th Jan 2021

Proration from 17th Jan 2021 - 24th Jan2021 = 8 days
Monthly Unit price/30 x 8 x quantity
plus 11 whole months

Example 4: Header Dates
Start 23rd Apr 2020
End 22nd Apr 2021

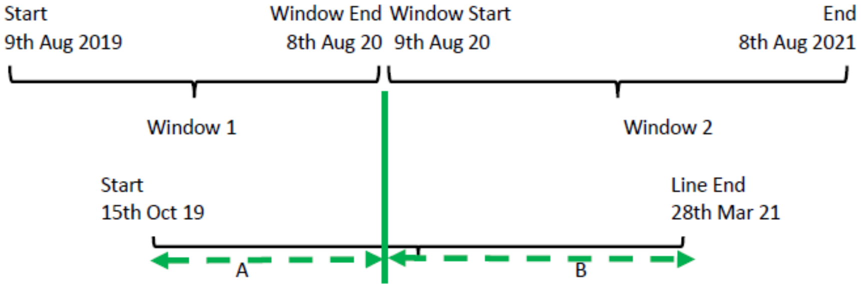
Line Item
Start 10th Jun 2020
End 13th Sep 2020

A Proration from 10th Jun - 22nd Jun = 13 days
Monthly Unit price/30 x 13 x quantity

B Proration from 23rd Aug - 13th Sep = 22 days
Monthly Unit price/30 x 22 x quantity

plus 2 whole months

Example 5: Header Dates - Multi year - with the Order Recognition Windows added
Start 9th Aug 2019 End 8th Aug 2021



Line Item
Start 15th Oct 2019
End 28th Mar 2021

Proration for Window 1
A Proration from 15th Oct - 8th Nov = 25 days — line item start to start of next month
Monthly Unit price/30 x 25 x quantity
plus 9 whole months

Proration for Window 2
B Proration from 9th Mar - 28th Mar = 20 days — Month start to line item end date
Monthly Unit price/30 x 20 x quantity
plus 7 months

8. Benefits

1. DOR reporting within ~4 hours post performing complex logic required for Order/Service Contract recognition.
2. QlikSense feature of QVD table creation and Set analysis (for deriving the columns) will competitively increase the processing speed by 90% for the reporting.
3. Multiple reports like DOR log, Missing DOR, Summary, Header, Line-item reports etc. can be created as per the Business requirement over the single DOR Dimension table in QlikSense or any equivalent reporting application.
4. View of all the transactions with having the granularity maintained up to debook/rebook/original entries per window of Order/Service Contract.
5. Daily scheduled reprocessing job will make sure of timely recognition of Order/Service Contract post meeting the predefined conditions and EORD, without being waiting for new transaction to process the DOR logic.
6. Data automation can be controlled as per Business requirement through scheduled workflows/jobs.
7. Support Sales Compensation for credit calculation and understand the future revenue by making derived recognized records readily available.
8. Flexible to support any enhancement as per upcoming Business requirement.

Please find below statistics explaining the details on improvised processing time and saving of manual efforts per execution of report.

System	Number of records	Memory Consumption	Time taken for Automated run	Time taken for Manual run	% Reduction in processing time	Manual work saved per execution
HIVE	2 M	50 GB	81 min	300 min	73%	219 min
QlikSense	2 M	20 GB	30 min	280 min	90%	250 min

9. Accomplishments and Lessons Learnt

1. By implementing this solution, we have successfully able to report on Order/Service Contract recognition within the required time frame of ~4 hours.
2. No synchronizing and combining the multiple ingested files are required which will eventually save ~1 hr of processing time.
3. Spark SQL is used for optimized execution of the complex logic.
4. Parallelism mechanism is used while running the Spark job which make sure every partition task gets single core for processing.
5. Optimized Row Columnar (ORC) file is used which improves the performance while reading, writing, and processing the data.
6. Intermediate temp tables (staging tables) are created to store the data temporarily while performing the complex DOR logic sequentially which distributes the load on memory by avoiding long running jobs or job failure post long run.
7. Improvised user experience by providing all aspects of the DOR reports required for Business process and additionally supporting downstream systems for performing Sales credit calculation and forecasting the revenue.

This communication contains general information only, and none of Deloitte Touche Tohmatsu Limited, its member firms or their related entities (collectively, the “Deloitte Network”), is, by means of this communication, rendering professional advice or services. Before making any decisions or taking any action that may affect your finances, or your business, you should consult a qualified professional adviser. No entity in the Deloitte Network shall be responsible for any loss whatsoever sustained by any person who relies on this communication.

used in this document, “Deloitte” means Deloitte Consulting LLP, a subsidiary of Deloitte LLP. Please see www.deloitte.com/us/about for a detailed description of our legal structure. Certain services may not be available to attest clients under the rules and regulations of public accounting.

Designed by CoRe Creative Services. RITM0457353