Pending Issues

1. How to multiplex streams – 100s of tables to process - how to add 10 logical tables in each stream?

2. Lets say we have olist_order_items files, and we have 100 of them how can we say load multiple files same time else only 1 file at a time ? Because if multiplexes and loads many files same time - the sequence will be messed up. Have you tried this ?

We should able to define an order at which tables must be consumed 2018 then 2019,then 2020…

If files are named with DDMMYYYY (datetimestamp) - cant we have any keywords to load them in that fashion ?

Python or bash touch function to modify the timestamp of the files in sequential order so Stru streaming can pick that time

SourceArchiveDir - to be used to move the file once processed

Checkpoint dir to be cleared for that Table only , directory was child of table directory

3. Can Spark streaming look for files recursively ?

Yes , fileNameOnly: whether to check new files based on only the filename instead of on the full path (default: false). With this set to `true`, the following files would be considered as the same file, because their filenames, "dataset.txt", are the same:

"file:///dataset.txt"

"s3://a/dataset.txt"

"s3n://a/b/dataset.txt"

"s3a://a/b/c/dataset.txt"

Pending Issues

4. How do we delete a partition and reprocess data for a day ?

Like data didn't arrive for yesterday, last week, last month and its arrived now ?

>> Sequence is important - Clean DB on ingestion time and upload all data again. Delta table could be rolled back to the old version easily.

5. Add sequence in every table records so we can order them

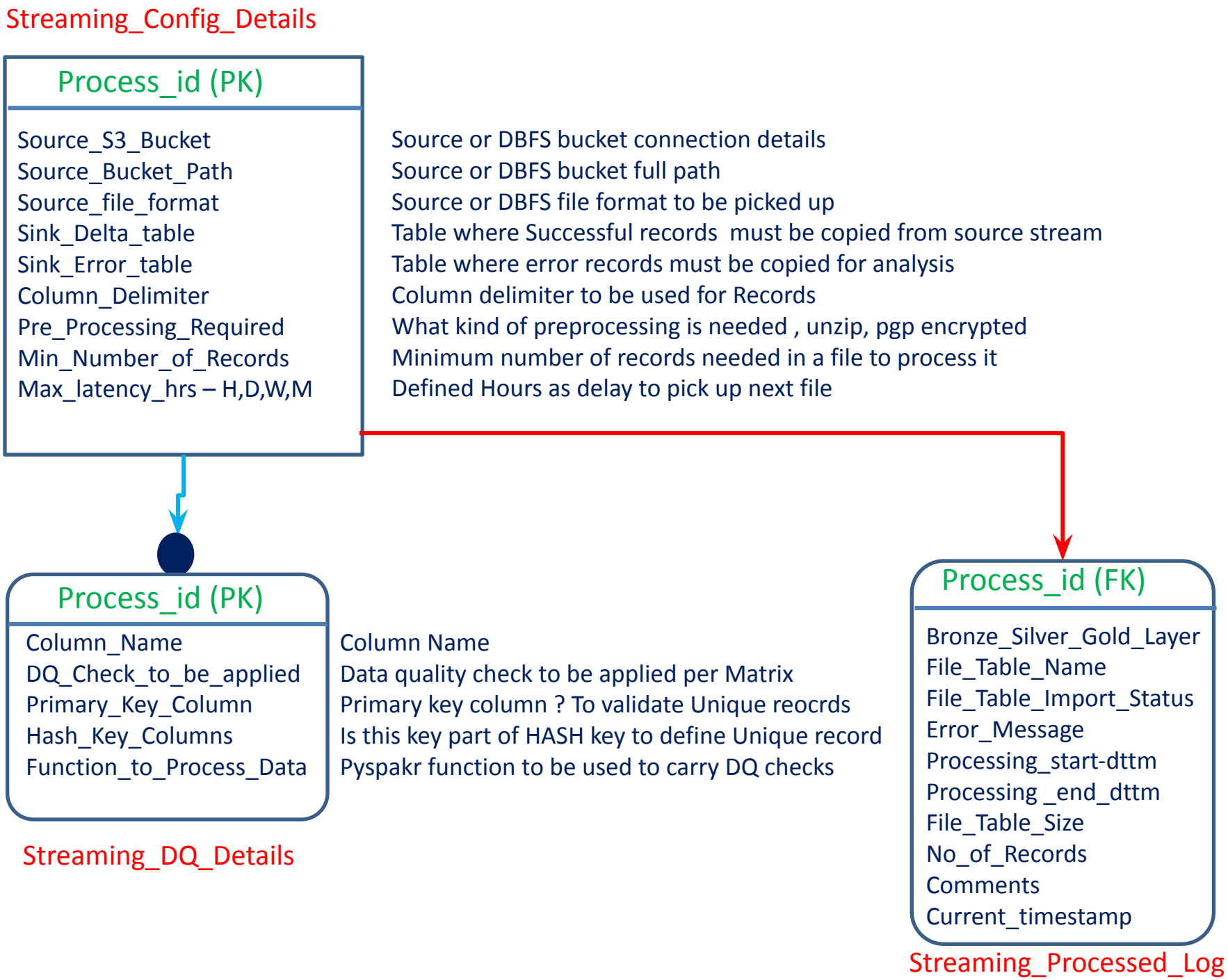
https://community.cloud.databricks.com/?o=7084176012667317#notebook/3330037115423360/command/3330037115423370

Drop Primary or duplicate key

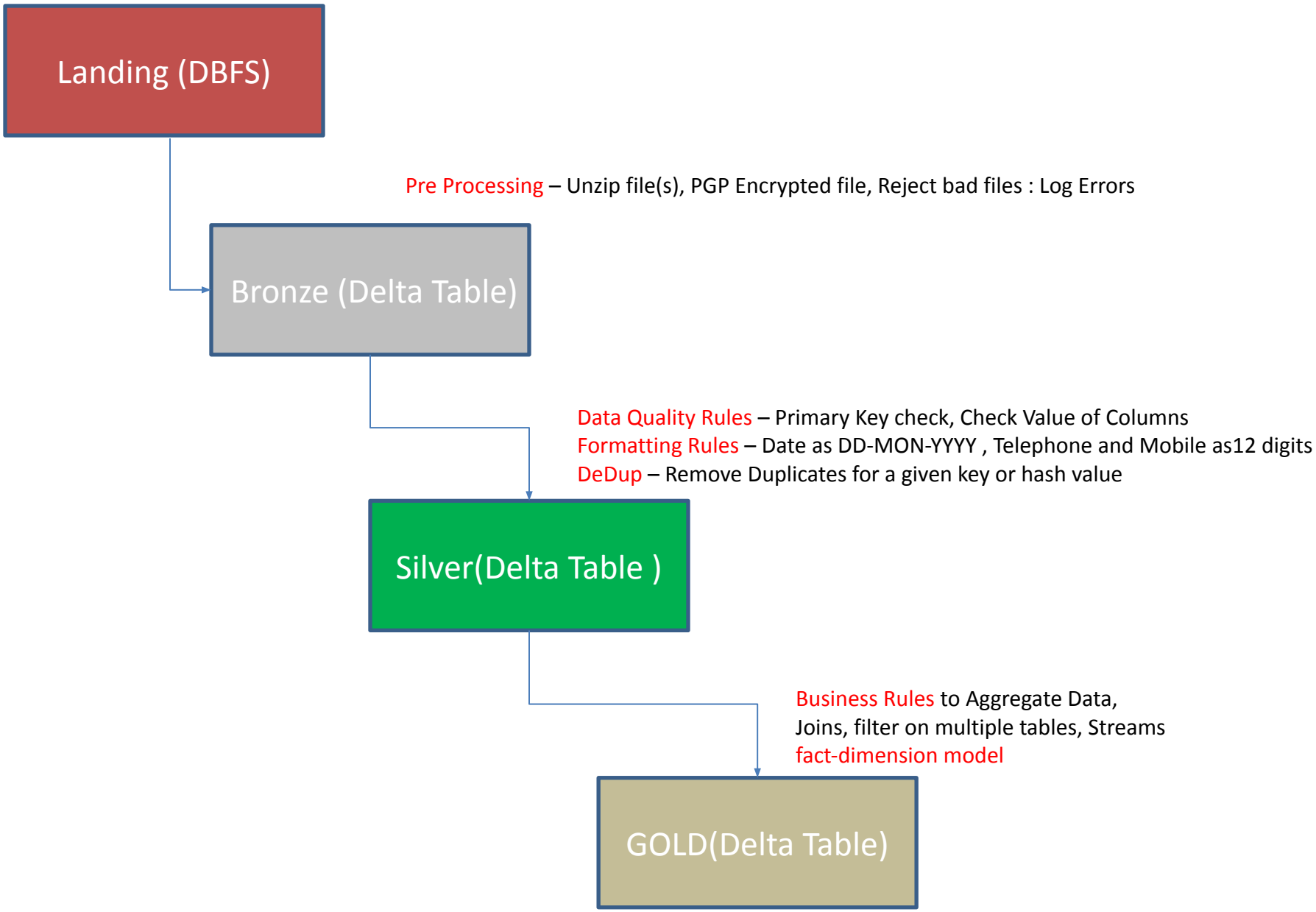
Use function - row_number() over(partition by <id> order by run_datetime DESC)=1

Issues

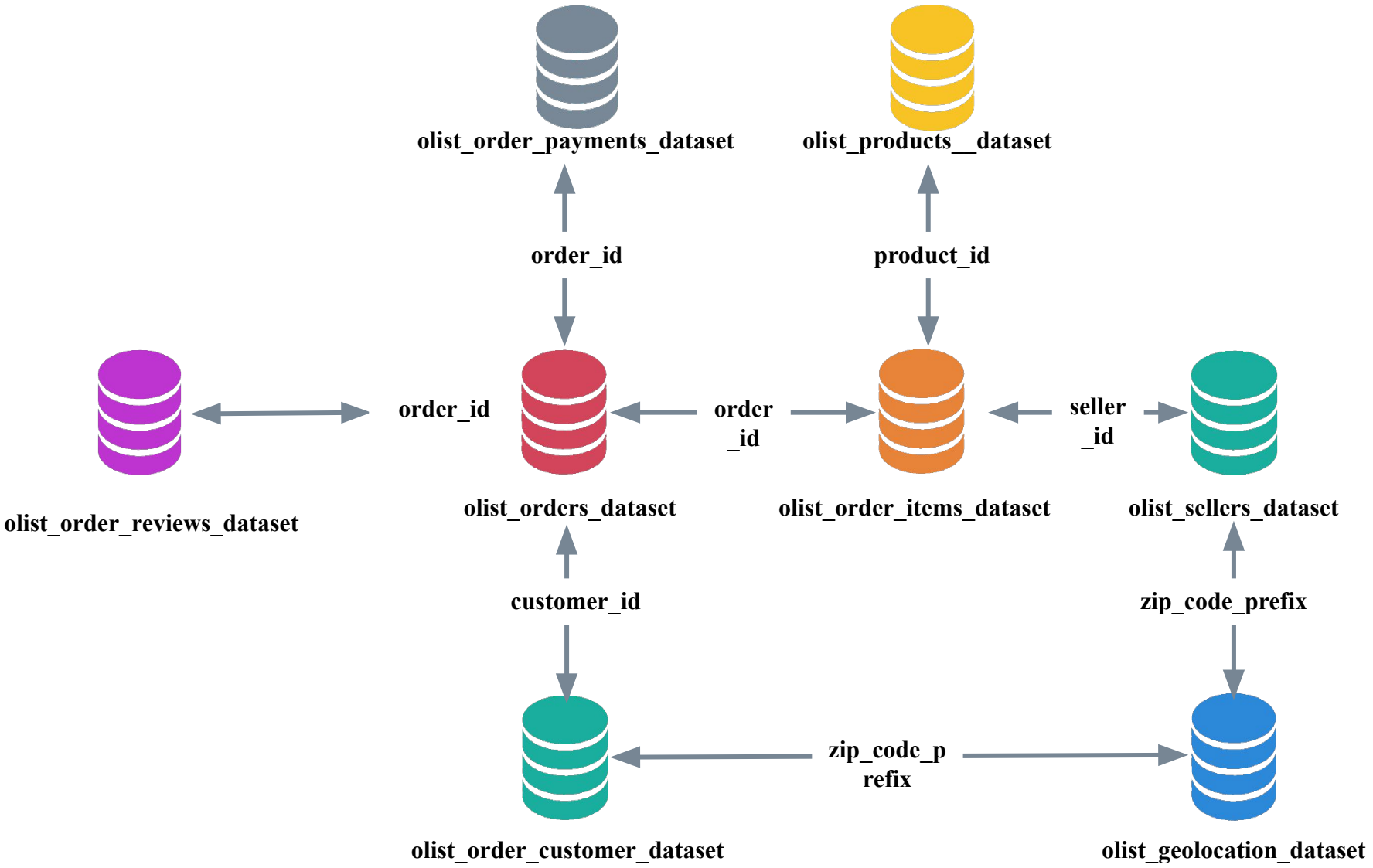
- Late arriving Data (watermark)
-


Streaming_Config_Details
Process_id (PK)
Source_S3_Bucket
Source_Bucket_Path
Source_file_format
Sink_Delta_table
Sink_Error_table
Column_Delimiter
Pre_Processing_Required
Min_Number_of_Records
Max_latency_hrs – H,D,W,M
Source or DBFS bucket connection details
Source or DBFS bucket full path
Source or DBFS file format to be picked up
Table where Successful records must be copied from source stream
Table where error records must be copied for analysis
Column delimiter to be used for Records
What kind of preprocessing is needed , unzip, pgp encrypted
Minimum number of records needed in a file to process it
Defined Hours as delay to pick up next file
Process_id (PK)
Column_Name
DQ_Check_to_be_applied
Primary_Key_Column
Hash_Key_Columns
Function_to_Process_Data
Column Name
Data quality check to be applied per Matrix
Primary key column ? To validate Unique reocrds
Is this key part of HASH key to define Unique record
Pyspakr function to be used to carry DQ checks
Streaming_DQ_Details
Process_id (FK)
Bronze_Silver_Gold_Layer
File_Table_Name
File_Table_Import_Status
Error_Message
Processing_start-dttm
Processing _end_dttm
File_Table_Size
No_of_Records
Comments
Current_timestamp
Streaming_Processed_Log

Tools - Autoloader +Delta Tables + Unity Catalog



Sample OLIST Dataset



We start with these 2 files first and then grow to other tables , I have ample of Datasets

Broze & Silver : OLIST Tables

olist_order_items_dataset

order_id
order_item_id
product_id
seller_id
shipping_limit_date
price
freight_value

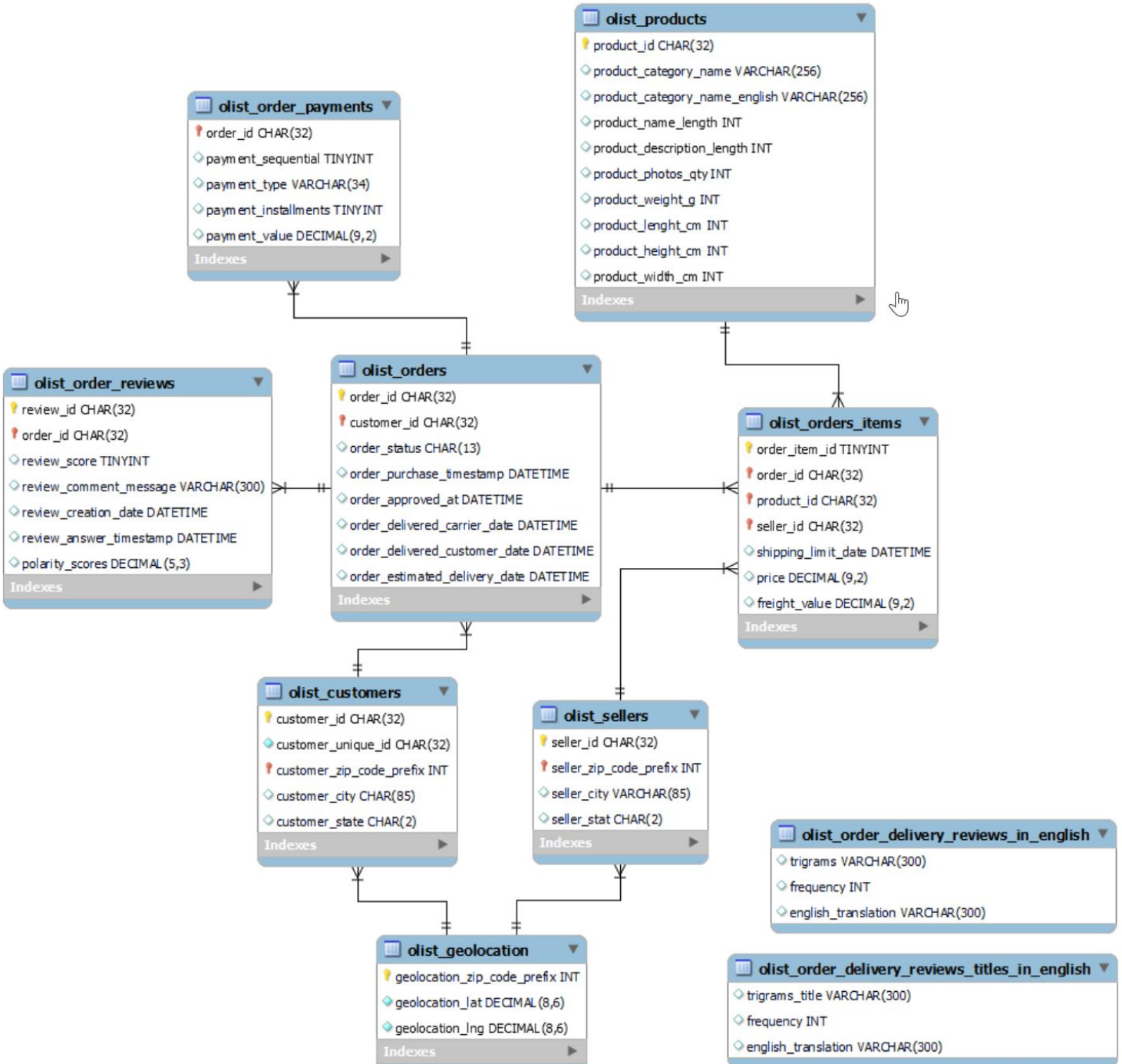
Aggregate products , total price per hour in stream
Select count(product_id), Sum(Price) from olist_order_items_dataset group by product_id

olist_orders_dataset

order_id
customer_id
order_status
order_purchase_timestamp
order_approved_at
order_delivered_carrier_date
order_delivered_customer_date
order_estimated_delivery_date

- help make better

https://medium.com/swl
-data-story-e53eb8e16f



Notebook details -

Workspace - vs_unified_streaming-Test

Cleanup - Notebook to cleanup the checkpoints and data

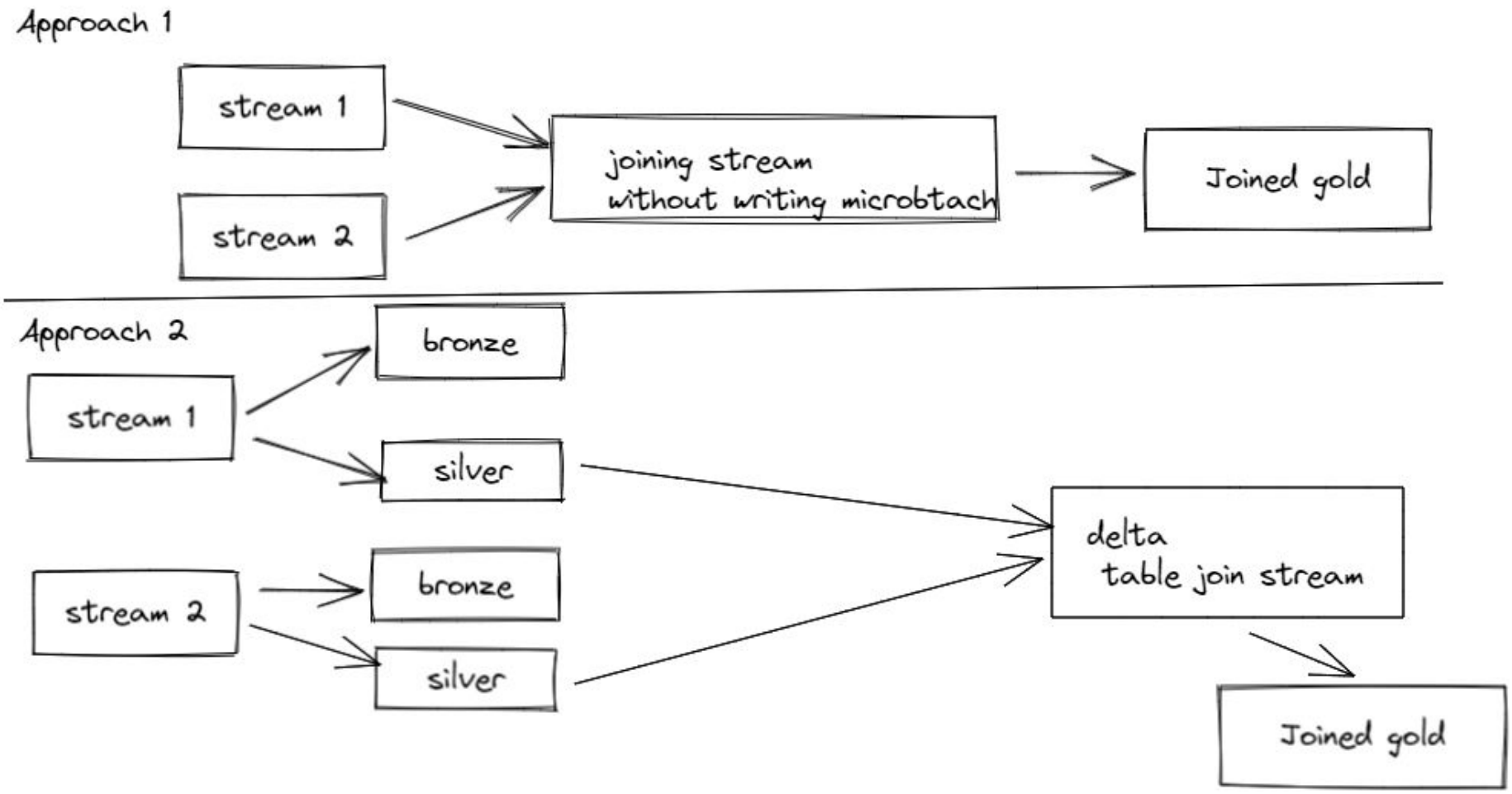
Create_setup_tables - Notebook to create setup tables - streaming_config_details, streaming_dq_config, streaming_log

Unified_delta_writer - notebook contains for each batch logic for bronze , silver and gold(aggregated)

Unfied_streaming - notebook contains the base class to create instances of each stream as configured in the create_setup_table

Stream-joins - to trigger multiple streams in parallel and also to perform stream joins.

Stream to Stream join design patterns



SILVER Table	Columns	DQ Rules to apply	Status
Data Quality Rules	Order_id Customer_id	Remove strange ASCII & any funny characters except Numbers and Alphabets	NA
	Order_id	Cant be NULL as It's a Primary Key	Done-Added primary key check and null constraint
	Order_status	Must have values “”,””,””,””	
	Price	Must be above 0.00 and non negative	NA
Formatting Rules	shipping_limit_date, order_purchase_timestamp, order_delivered_carrier_date , order_delivered_customer_date , order_estimated_delivery_date	Format Date as DD-MON-YYYY HH24:MI:SS , Format Timestamp as DD-MON-YYYY HH24:MI:SS	Done-Created a separate function to convert date in DD-MON-YYYY HH24:MI:SS
De-Duplication	Order_id	Always add UNIQUE value when inserted at Silver table. Join the target table with input stream to see if any duplicates	Done-Upsert statement impleneted

Streams	Criteria	Result and Analysis	Status
Aggregation within The Same Stream	Aggregate the ORDERS and sum of REVENUE in an hour using window and watermarking Table : Columns:	Implemented it using microbatch approach for each stream. Grouping is calculated for each batch and merged using upsert into aggregated table.	Done
	Aggregate ITEMS in an hour using window and watermarking		NA
Joining 2 Streams and Aggregating with Window function	Add same day files for Orders and Items and then join by ORDER_ID	We could aggregate successfully from SIlver data but we cant UPSERT the Gold table thru stream. Limitation - Foreachbatch is not functioning with stream join , Solution - Proposed solution is to append into a temp table. A new stream will read the data from this table and upsert into the final joined table.	Done
Joining Stream data from Silver with static GOLD tables again used as	Join 2 streams and GOLDEN **table** for the aggregated results	Scenerio 1. Inner join of items & order son Order_id -> If you use watermarking for each batch can't be used with 2 streams 2. Aggregation can work between streams and carry upsert on GOLD table but simple join	Done

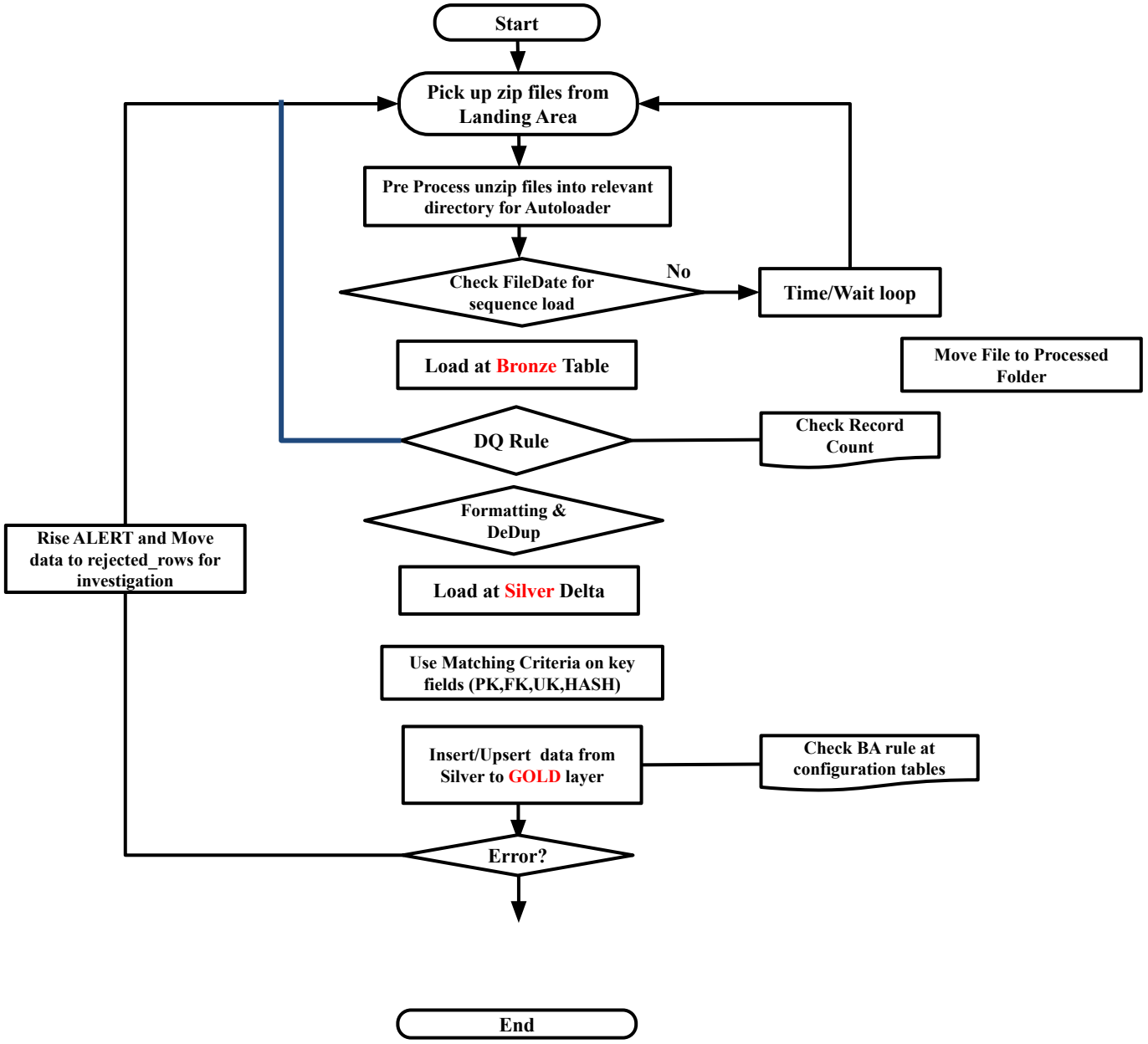
GOLD Aggregations	Query	Status
What is the number of customers by state?		
What is the number of orders by month?		
What are the top 5 product categories?		
		pending
		DONE

Test cases -

Bronze	Folder structure for files	Needs to be done in source directory
Bronze	Table schema Checked	Schema enforcemnet needs to be added
Bronze	PK/FK/Unique key details available	Yes
Silver	Schema validation of tables between source and Target	Schema enforcemnet needs to be added
Silver	Load status validation for runs	Yes
	Duplicate checks(PK and Hash key)	Upsert - Yes
Silver	Null checks (PK, Hash key or Audit keys)	Yes- implemented for primary keys
Silver	Date checks (ingested date and insert/update Date)	Yes - record insert date is being captured

Test cases Contd.

Silver	record status	Yes - record status as valid and invalid
Silver	Reconciliation checks for distinct PK, count of files and count of Input	Not imeplemented
GOLD	Join conditions	Yes -Testing in progress
GOLD	aggregations	Yes- gold aggregation tables are being created


Start
Pick up zip files from Landing Area
Pre Process unzip files into relevant directory for Autoloader
Check FileDate for sequence load
No
Time/Wait loop
Load at Bronze Table
Move File to Processed Folder
DQ Rule
Check Record Count
Formatting & DeDup
Rise ALERT and Move data to rejected_rows for investigation
Load at Silver Delta
Use Matching Criteria on key fields (PK,FK,UK,HASH)
Insert/Upsert data from Silver to GOLD layer
Check BA rule at configuration tables
Error?
End

Next Piece of Work

- Great_expectation framework to be embedded
- Analysis of DQ and log table
- Autoloader implementation
-


Read S3 bucket data into landing Bucket
Migrate the data into Bronze Bucket
Apply the Data cleansing Rule(s)
Copy Data into Silver Bucket
Apply Business Rules
Process Data into GOLD table

Logging

All log entries (INFO, DEBUG, and DETAIL) should be placed in a log table located on Delta Database. The elements on each line of the resulting log file should have:

- ‘File_name’
- ‘Table_name’
- Date and Time
- The name of the instrumented action
- Type of Error
- Error Text

User Interface

DWH provide no GUI user interface. Access to DWH database is made through SQL thick client

DWH SQL interface is used by developers, administrators, and other users to:

a Manage the design and configuration of DWH data

b Create and maintain DWH processes and process structures

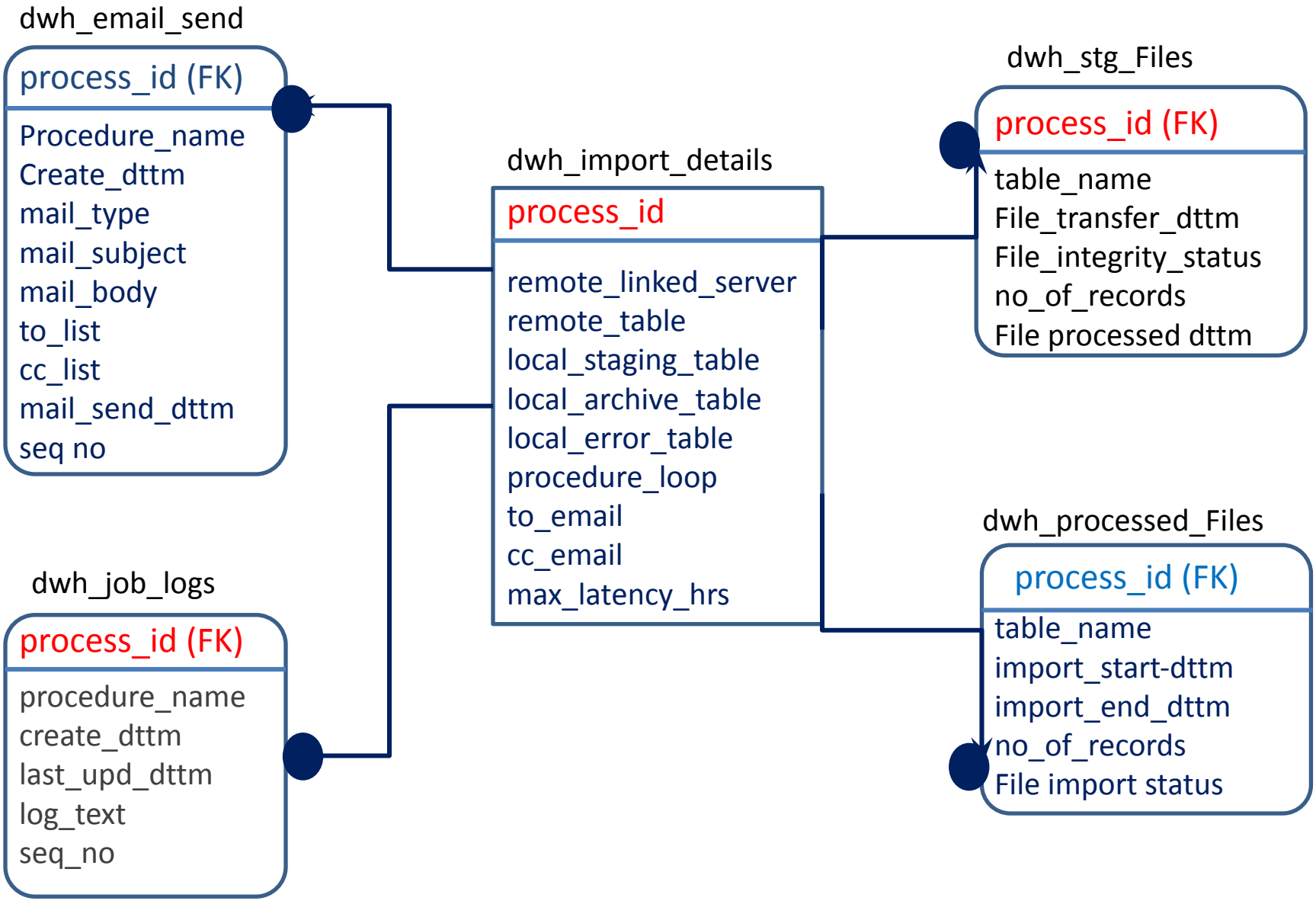
c Configure and maintain DWH

d Access data stored in the DWH database

Failed Jobs & Rerun

1) A failed job can be restarted from SQL scheduler
2) If a job fails at any stage before data moves to main SCV tables, it can be restarted

Appendix


dwh_email_send
process_id (FK)
Procedure_name
Create_dttm
mail_type
mail_subject
mail_body
to_list
cc_list
mail_send_dttm
seq no
dwh_job_logs
process_id (FK)
procedure_name
create_dttm
last_upd_dttm
log_text
seq_no
dwh_import_details
process_id
remote_linked_server
remote_table
local_staging_table
local_archive_table
local_error_table
procedure_loop
to_email
cc_email
max_latency_hrs
dwh_stg_Files
process_id (FK)
table_name
File_transfer_dttm
File_integrity_status
no_of_records
File processed dttm
dwh_processed_Files
process_id (FK)
table_name
import_start-dttm
import_end_dttm
no_of_records
File import status

Monitoring

Steps to Monitor Staging Load Process

1) For any issue with import job (data is corrupt, number of records don't match, constraint issue etc.), an error alert to be generated and sent to support team.

2) For each data File maximum latency can be set up in a reference table. Once it passes that latency a warning mail needs to be issued. (For example we can set up maximum latency for a File as 24 hours. If a new File doesn't arrive after 24 hours since last File was received, a warning mail to be issued). This warning should contain File name and last time it was received.

3) If there is a problem with a particular File, that File should be moved to an error area. The process will not stop and will continue with the next available File.

4) If a File needs to be reprocessed (previously it was supplied with partial data) then corresponding entries need to be deleted from control tables and only then it can be processed by automated import job.

5) In case a File fails due to data issue (moving data from file to staging table) then staging table can be queried to find out the issue. Staging table will be truncated in next run following a successful import of File

Staging Table

Core Table FactNLTRANS

Exception Table NLTRANS_Exception

OpenAccounts will populate oalive60 database at Remote server every day, SQL to download daily data and call "Staging.dbo.nltrans_load_proc" procedure.

Stored Procedure "Staging.dbo.nltrans_load_proc" will perform following

- Call Dwh_Import.dbo.log_insert to insert the entry into Dwh_Job_Logs
- Procedure will check whether the data has been processed before successfully
- Procedure will exit successfully and pass the output as 0 if the data has already : processed
- Procedure will create an entry in Dwh_Processed_Files table to indicate : Processing has started
- Procedure will truncate the Staging tables
- Procedure will import the data into Staging table from linked server connection
- Procedure will update the Dwh_stg_Files table and set the : File_Integrity_Status to "F" to indicate import is failed
- Procedure will check records loaded into Staging are equal to number of records available at source server. Any difference, procedure will fail and update the Dwh_Processed_Files table accordingly
- Procedure will filter all duplicate records and insert into exception table in Core
- Procedure will Remove all duplicate records from Staging table
- Procedure will update Dwh_Import_Files table and set the File_Import_Status to "F" for any failures while loading the data into Core tables
- Update Dwh_Processed_Files table and set File_Import_Status to "S" and Import_End_Dttm to GetDate() after loading the data successfully into Core table
- In case of any failure, Procedure will raise alerts to support email

Standard UDF Procedures

UDF to write entry into a a log table - Streaming_Job_Logs

Name	Type	Input Parameter	Mandatory	Type
log_insert_details	UDF	Process_ID	Y	varchar(30)
		Procedure_Name	Y	varchar(30)
		Log_Dttm	Y	Datetime
		Log_Text	Y	varchar(200)

Mail_insert_proc

This procedure will be used to send emails. This will create an entry in a database table.

Name	Type	Input Parameter	Mandatory	Type
mail_insert_proc	Procedure	p_Process_ID	Y	varchar(30)
		p_Mail_Type	Y	varchar(1)
		p_Mail_Subject	Y	varchar(50)
		p_Mail_Body	N	varchar(200)
		p_To_List	Y	varchar(200)
		p_CC_List	N	varchar(200)

Glue Code Snippet

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.dynamicframe import DynamicFrame
args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)


source_s3_path = 's3://source-bucketv1/source data/Customer.csv'
landing_s3_path = 's3://landing-bucketv1/Targetdata/Customer/'
# Script generated for node S3 bucket
df = spark.read.option("header", "true").csv(source_s3_path)


#df = S3bucket_node1.toDF()
df.show()
S3bucket_node = DynamicFrame.fromDF(df, glueContext,"dynamicdf")
# Script generated for node S3 bucket
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
frame=S3bucket_node,
connection_type="s3",
format="glueparquet",
connection_options={
"path": landing_s3_path,
"partitionKeys": [],
},
transformation_ctx="S3bucket_node3",
)
job.commit()
```

DWH Design

1	What happens if we miss a File for some reason at nighty build?	OpenAccounts tables are very tightly linked . For example we may have an entry at sales ledger/purchase ledger and it should be at NLledger(approved) else Nlunposted (unapproved) but it won't be due to missing data. What should happen in case we don't extract records for a File in nightly run? This is pretty hard issue to resolve. Should ETL stop entire build process or process it and leave data in whatever state it is until next run to correct itself? I can surely alert to warn ops team. **It was decided to stop ENTIRE ETL PROCESSING and raise alert**
2	Primary Keys at Source	Primary key or combination of keys to make every record unique at each tables. This is important to locate duplicate records and supress them at staging. The Core or DWH DB will be linked with surrogate keys. They are separate than these source data keys. Used PROGRESS_RECID & PROGRESS_RECID_IDENT_ to identify Unique records at each File
3	Change Capture	SQL server CDC in case we decide to use for extracting deltas will need enterprise license. There is an alternative for std edition https://standardeditioncdc.codeplex.com/ , I am not sure what VW policies are to use freeware scripts. Sliding 1 month Window to load data for a month But how to locate months data? Load all lookup tables every day for a month
4	Dimension Value Alterations	I have used Type 2 dimension using date values for all dimensions. This is done as there were no clear details how often dimension values will change. Type 2 can take care of changing values and in case values don't change that often it's not going to cause any breakdown else slow response
5	Record count difference (Source and Staging)	STOP ENTIRE PROCESSING and raise alerts
		Dimension Type ■ Supplier /Customer history TYPE 2