



## Ingestion and historization in the Data Lake

Illia Todor

corporate.hrs.com

#### /iamtodor



# BRIGHTUROVE

Data Engineer

AWS Cloud Practitioner Certified

#### /iamtodor



# BRIUHTUROVE

Data Engineer

AWS Cloud Practitioner Certified

#### HRS GROUP

#### /iamtodor



## BRIGHTUROVE

Data Engineer

AWS Cloud Practitioner Certified

#### HRS GROUP

https://github.com/iamtodor https://stackoverflow.com/users/5151861/iamtodor https://iamtodor.medium.com/ https://www.linkedin.com/in/iamtodor/



## provides the world's premier lodging experience to corporations globally.







1,000 employees



yearly contacts with Hotels



## Lodging as a Service

SMART

Gree stay

7 HRS clean & Safe

CONTINUOUS PROCUREMENT

SMART BOOKING

TOUCHLESS STAY

TRANSIENT | MEETINGS | GROUPS | LONGSTAY

Part Part Part Part





## Lodging as a Service





SMART

Green Stay

2 HRS clean & Safe

CONTINUOUS PROCUREMENT

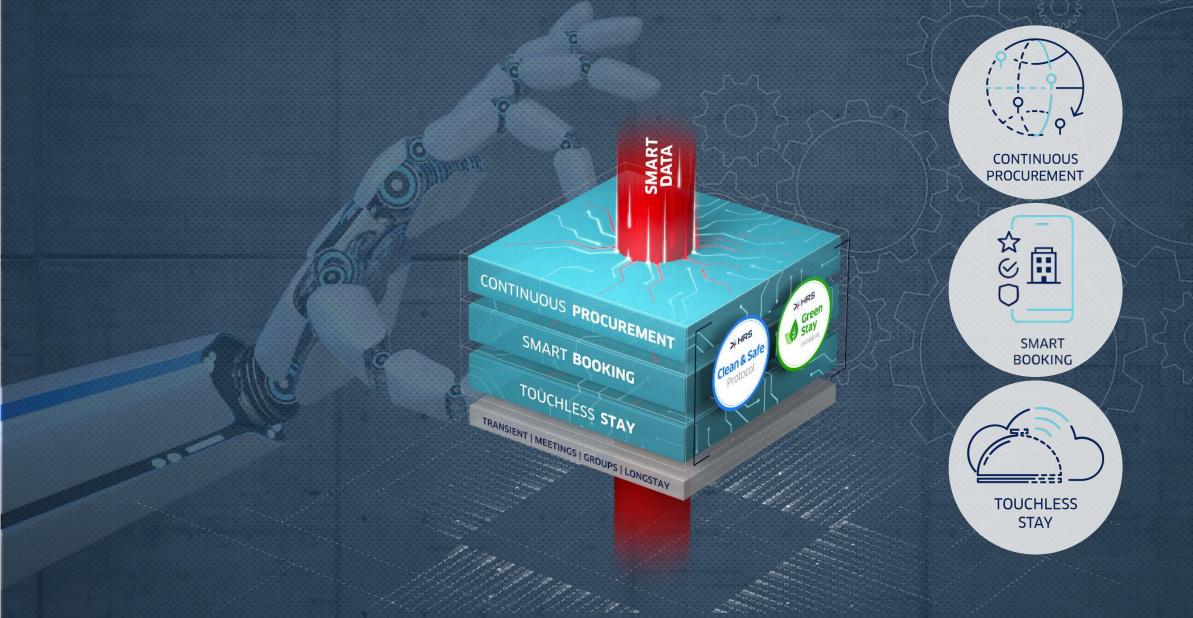
SMART BOOKING

TOUCHLESS STAY

TRANSIENT | MEETINGS | GROUPS | LONGSTAY

## Lodging as a Service





#### Request



- DataSource
- Historical Changes
- DataLake
- DataWarehouse
- Dashboards



#### Historization: the process



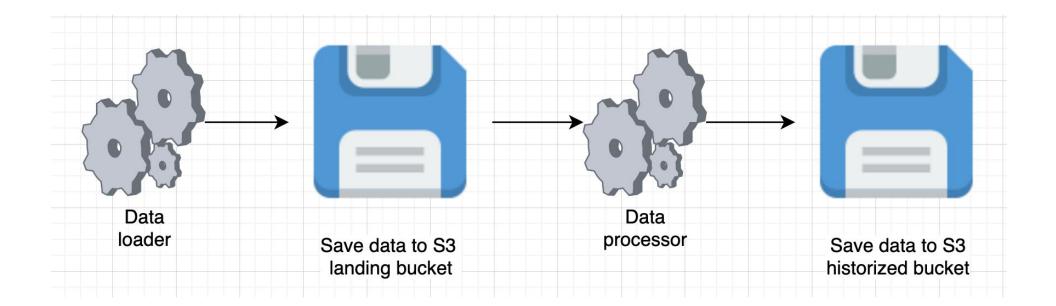
We wanted to move ingestions from DataSource into Data Lake along with applying historization approach. The data we need in the DWH for reporting.

The idea of historization approach is to track historical changes.

Operation in the source DB	Field value	valid_from	valid_to
Insert	Andrew	2021-01-01	2021-01-03
Update	Mark	2021-01-03	2021-01-05
Update	Daniel	2021-01-05	2099-12-31

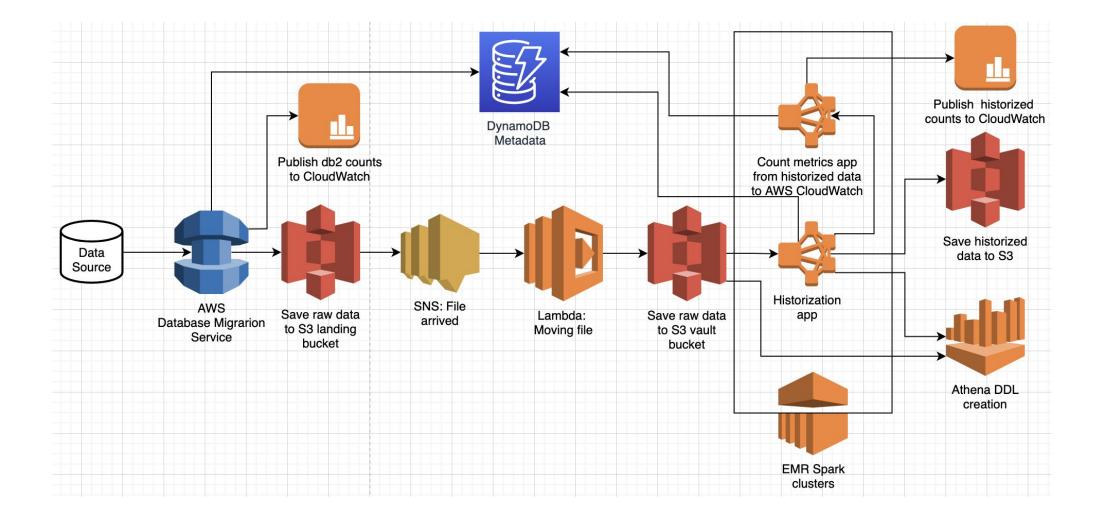
#### Architecture: abstract





#### Architecture: in details





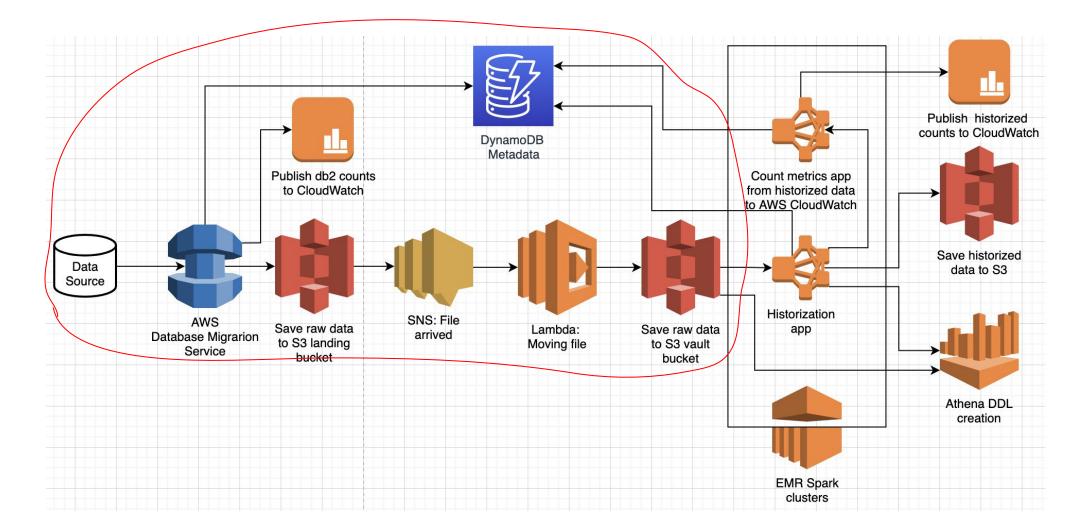
#### Infrastructure as a Code





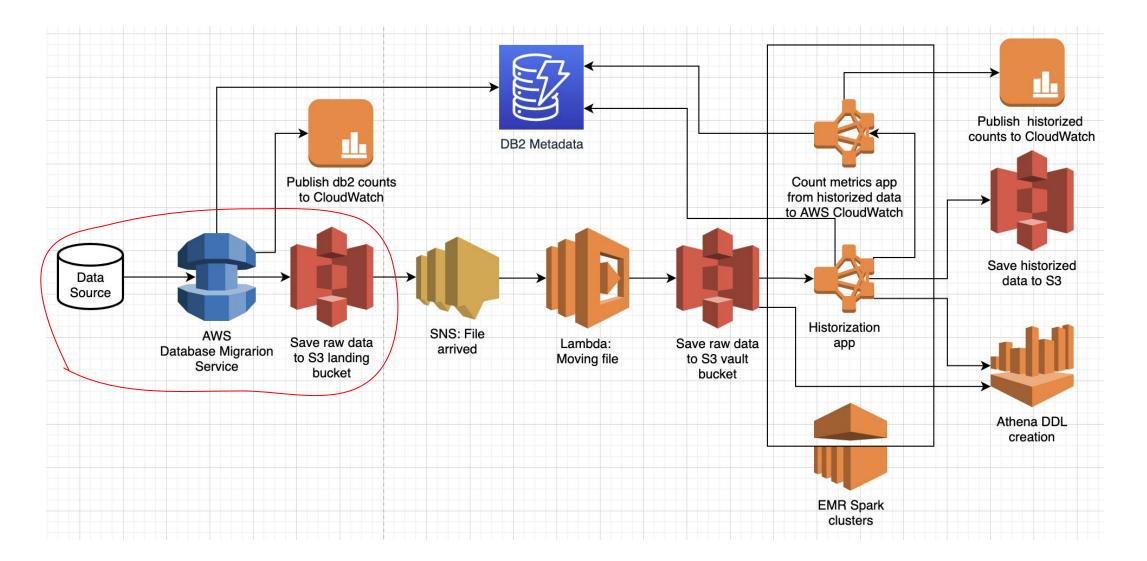
#### First part: global overview











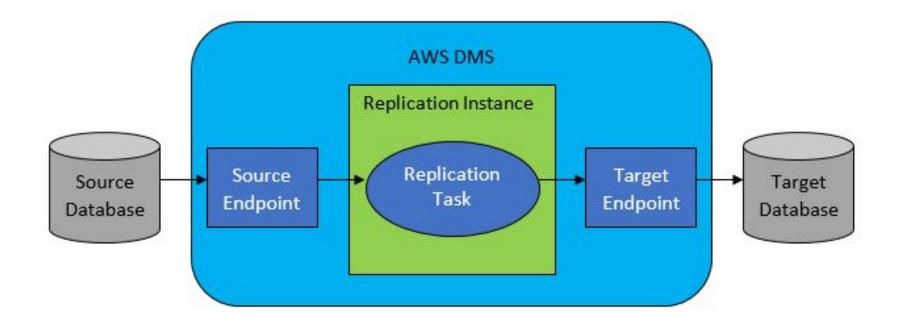
#### AWS DMS: Database Migration Service





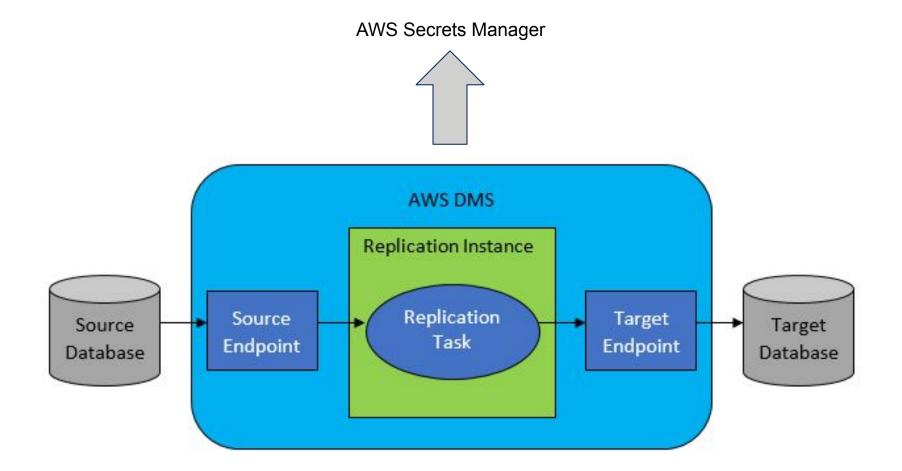
#### AWS DMS: Database Migration Service





#### AWS DMS: Database Migration Service





#### **DMS: Replication Instance**



Replication instance type	vCPU	Memory (GiB)
Memory Optimized		
dms.r4.large	2	15.25
dms.r4.xlarge	4	30.5
dms.r4.2xlarge	8	61
dms.r4.4xlarge	16	122
dms.r4.8xlarge	32	244
dms.r5.large	2	16
dms.r5.xlarge	4	32
dms.r5.2xlarge	8	64
dms.r5.4xlarge	16	128

### DMS: Target Endpoint



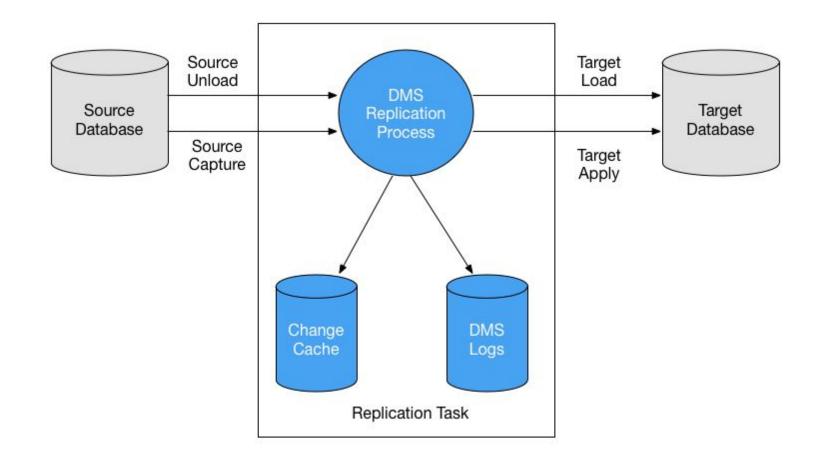
1 S	33ProdDMSEndpoint:
2	Type: AWS::DMS::Endpoint
3	Properties:
4	EndpointIdentifier: prod-s3-landing-parquet-endpoint
5	EngineName: s3
6	EndpointType: target
7	S3Settings:
8	BucketName: 'bucket_name'
9	BucketFolder: 'prod/prefix'
10	CompressionType: gzip
11	ExtraConnectionAttributes:
	<pre>dataFormat=parquet;datePartitionDelimiter=DASH;datePartitionEnabled=false;datePartitionSequence=YYY</pre>
Y	'MMDD; '

#### DMS: Target Endpoint



1	S3ProdDMSEndpoint:
2	Type: AWS::DMS::Endpoint
3	Properties:
4	EndpointIdentifier: prod-s3-landing-parquet-endpoint
5	EngineName: s3
6	EndpointType: target
7	S3Settings:
8	BucketName: 'bucket_name'
9	BucketFolder: 'prod/prefix'
10	CompressionType: gzip
11	ExtraConnectionAttributes:
	'dataFormat=parquet;datePartitionDelimiter=DASH;datePartitionEnabled=false;datePartitionSequence=YYY
	YMMDD; '







1	. ReplicationTask:
2	<pre>Type: AWS::DMS::ReplicationTask</pre>
4	<pre>ReplicationTaskIdentifier: !Sub "\${Env}-{source}-{target}-{schema_name}-{table_name}"</pre>
5	MigrationType: full-load
6	ReplicationInstanceArn: !ImportValue replication-instance-arn
7	SourceEndpointArn: !ImportValue source-endpoint-arn
8	
g	<pre>ReplicationTaskSettings: '{"Logging": {"EnableLogging": true}}'</pre>
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
4.7	



```
Type: AWS::DMS::ReplicationTask
      ReplicationTaskIdentifier: !Sub "${Env}-{source}-{target}-{schema_name}-{table_name}"
      MigrationType: full-load
      ReplicationInstanceArn: !ImportValue replication-instance-arn
      SourceEndpointArn: !ImportValue source-endpoint-arn
      TargetEndpointArn: !ImportValue s3-prod-endpoint
12
```



```
Type: AWS::DMS::ReplicationTask
     ReplicationTaskIdentifier: !Sub "${Env}-{source}-{target}-{schema_name}-{table_name}
     MigrationType: full-load
      ReplicationInstanceArn: !ImportValue replication-instance-arn
     SourceEndpointArn: !ImportValue source-endpoint-arn
      TargetEndpointArn: !ImportValue s3-prod-endpoint
10
          "rules": [
              "rule-id": 1,
16
18
```



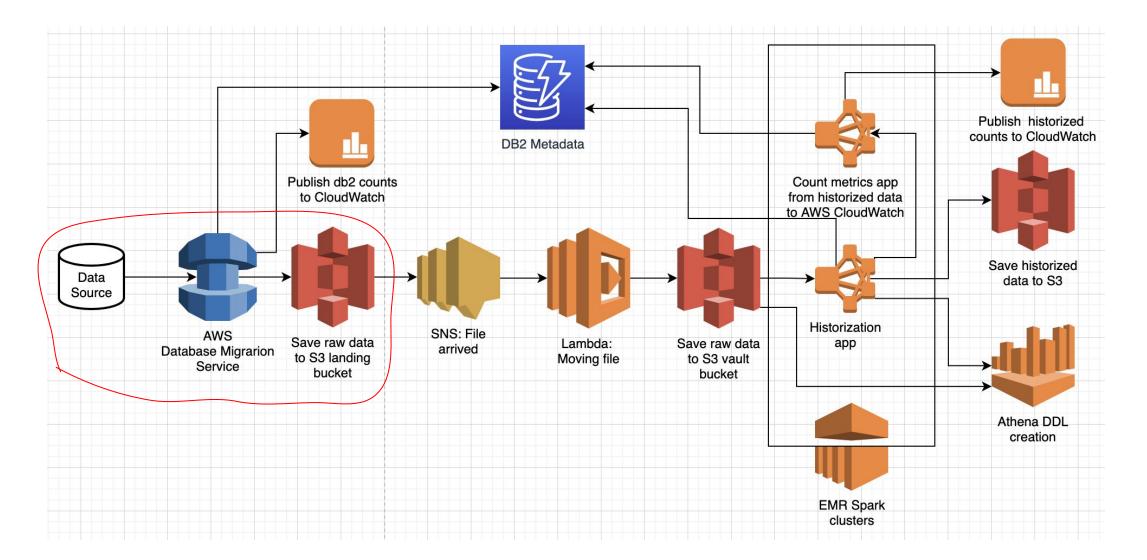
```
Type: AWS::DMS::ReplicationTask
      ReplicationTaskIdentifier: !Sub "${Env}-{source}-{target}-{schema_name}-{table_name}'
      MigrationType: full-load
      ReplicationInstanceArn: !ImportValue replication-instance-arn
      SourceEndpointArn: !ImportValue source-endpoint-arn
      TargetEndpointArn: !ImportValue s3-prod-endpoint
      ReplicationTaskSettings: '{"Logging": {"EnableLogging": true}}'
              "rule-type": "selection",
21
```



```
Type: AWS::DMS::ReplicationTask
      ReplicationTaskIdentifier: !Sub "${Env}-{source}-{target}-{schema_name}-{table_name}"
      MigrationType: full-load
      ReplicationInstanceArn: !ImportValue replication-instance-arn
      SourceEndpointArn: !ImportValue source-endpoint-arn
      TargetEndpointArn: !ImportValue s3-prod-endpoint
      TableMappings:
10
11
           "rules": [
12
13
14
               "rule-type": "selection",
               "rule-id": 1,
15
               "rule-name": 1,
16
17
               "rule-action": "include",
18
               "object-locator": {
                 "schema-name": "schema_name",
19
20
                 "table-name": "table_name"
21
22
23
24
```

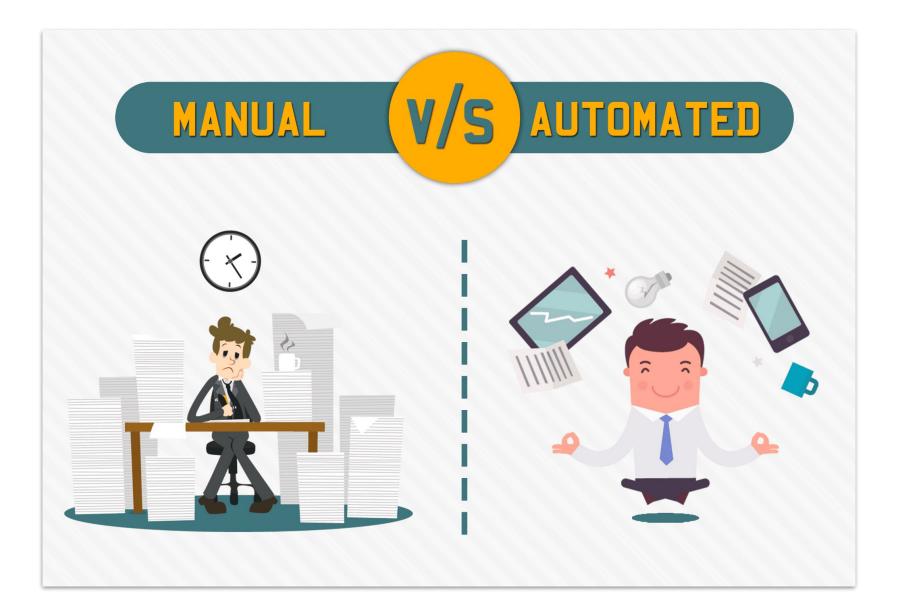






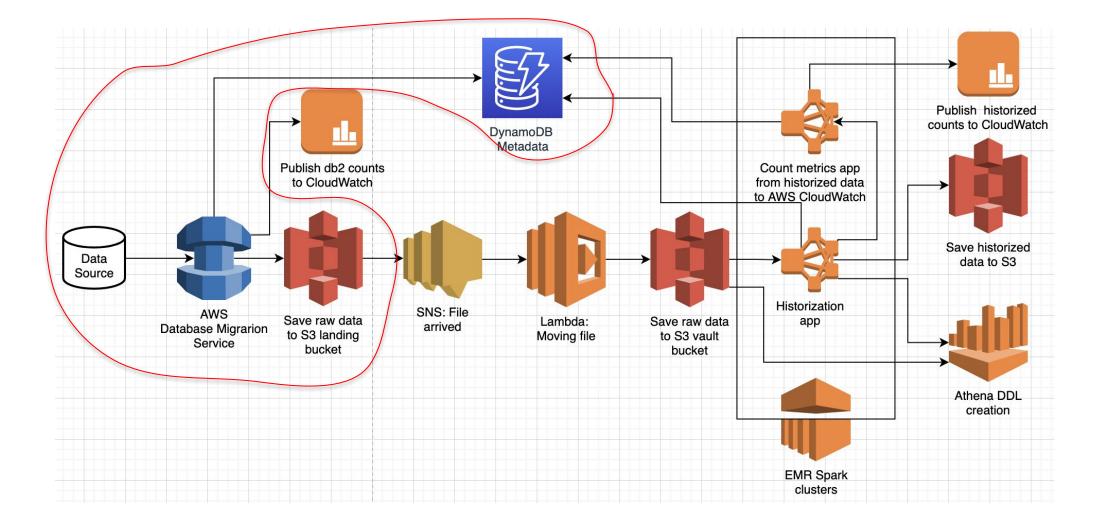
#### Process automation and scaling





#### First part: Metadata management





#### What is metadata itself





#### Suitable tool for metadata





CSV

#### Suitable tool for metadata





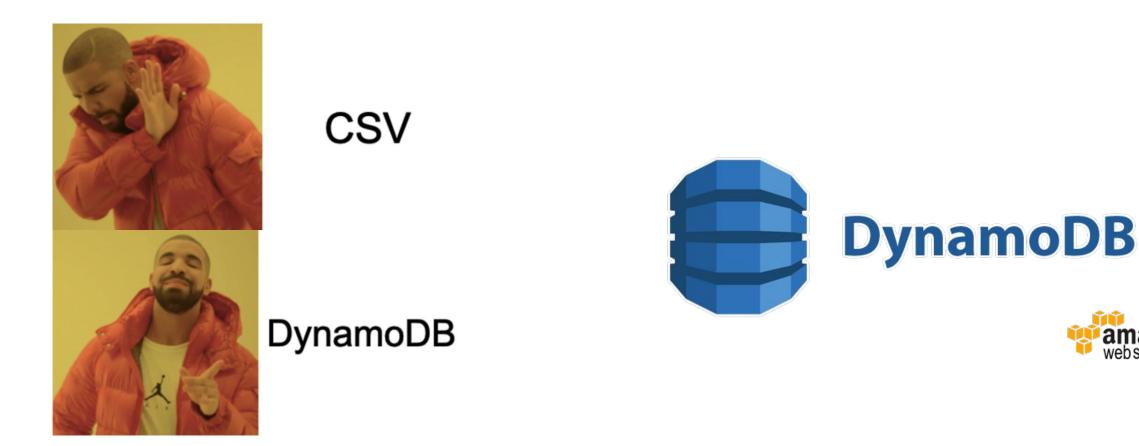
CSV

#### DynamoDB

#### Suitable tool for metadata



amazon webservices



### config.json



```
"SchemaName": "schema_name",
2
3
    "TableName": "table_name",
    "Source": "db2",
    "StartValue": "1970-01-01 00:00:00.000000",
    "EndValue": "2021-05-11 00:00:00.000000",
    "PrimaryKeyFields": "pk_field",
8
    "OutputFormat": "parquet",
10
11
    "IsAthenaTableRequired": true,
    "IsTargetPartitioned": false
15 }
```

### config.json







```
"SchemaName": "schema_name",
    "TableName": "table_name",
    "Source": "db2",
5
    "StartValue": "1970-01-01 00:00:00.000000",
6
    "EndValue": "2021-05-11 00:00:00.000000",
7
    "PrimaryKeyFields": "pk_field",
8
    "OutputFormat": "parquet",
10
12
    "IsAthenaTableRequired": true,
    "IsTargetPartitioned": false
15
```



```
"SchemaName": "schema_name",
    "TableName": "table_name",
 3
    "Source": "db2",
    "StartValue": "1970-01-01 00:00:00.000000"
    "EndValue": "2021-05-11 00:00:00.000000",
    "PrimaryKeyFields": "pk_field",
8
    "TimestampColumnName": "",
9
    "OutputFormat": "parquet",
10
11
    "IsAthenaTableRequired": true,
13
    "IsTargetPartitioned": false
15
```



```
"SchemaName": "schema_name",
    "TableName": "table_name",
    "Source": "db2",
    "StartValue": "1970-01-01 00:00:00.000000",
    "EndValue": "2021-05-11 00:00:00.000000",
8
    "PrimaryKeyFields": "pk_field",
    "OutputFormat": "parquet",
10
11
    "IsAthenaTableRequired": true,
    "IsTargetPartitioned": false
```



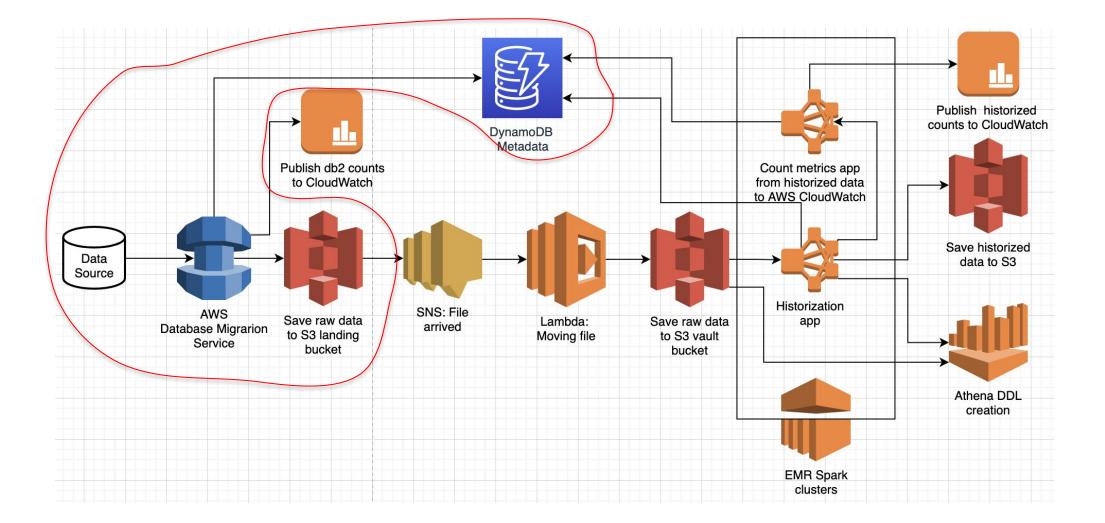
```
"SchemaName": "schema_name",
    "TableName": "table_name",
    "Source": "db2",
    "StartValue": "1970-01-01 00:00:00.000000",
 6
    "EndValue": "2021-05-11 00:00:00.000000",
    "PrimaryKeyFields": "pk_field",
 8
    "OutputFormat": "parquet",
10
    "BrokenDateTimeColumns": "",
11
    "DecimalWithNullColumns": "",
12
    "IsAthenaTableRequired": true,
    "IsTargetPartitioned": false
```



```
"SchemaName": "schema_name",
     "TableName": "table_name",
    "Source": "db2",
    "StartValue": "1970-01-01 00:00:00.000000",
    "EndValue": "2021-05-11 00:00:00.000000",
    "PrimaryKeyFields": "pk_field",
 8
     "OutputFormat": "parquet",
10
     "IsAthenaTableRequired": true,
13
     "IsTargetPartitioned": false
14
```

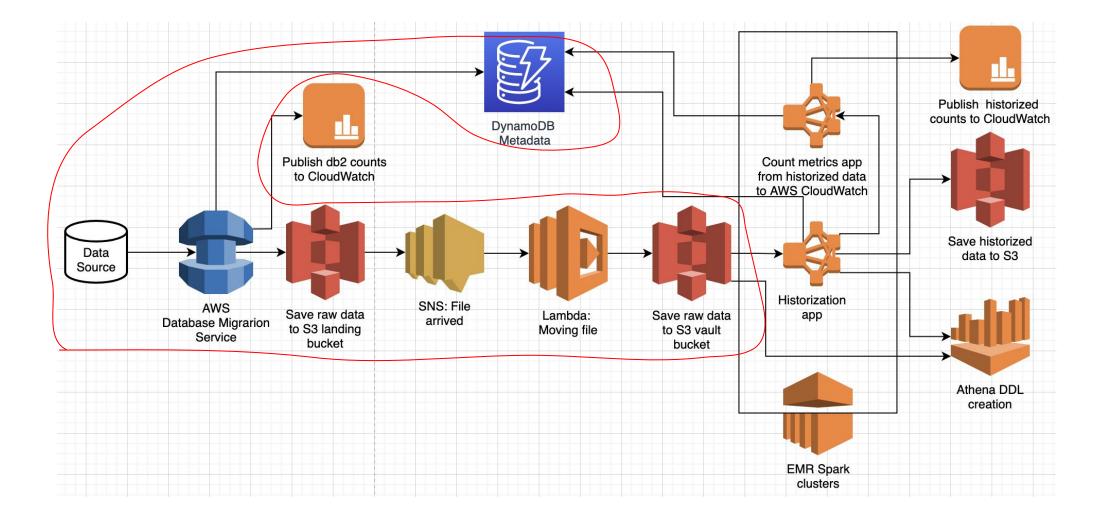
#### First part: Metadata management





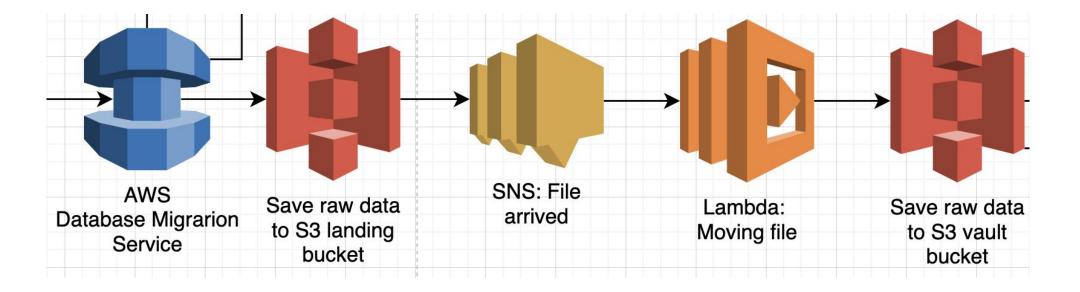
#### First part: DMS saving data





#### DMS: saving data





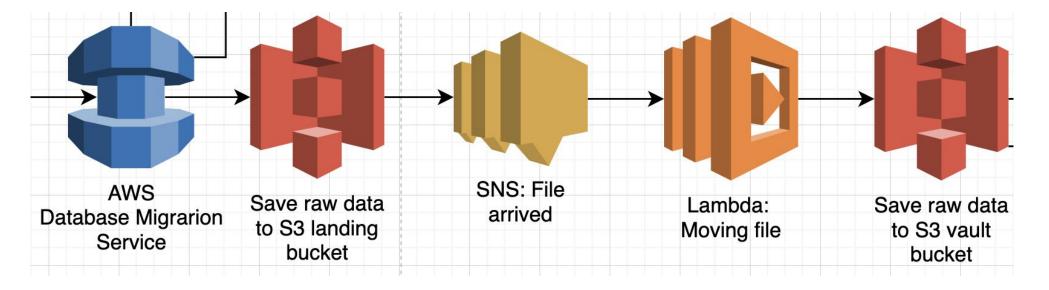
### DMS: saving data



AWS		SNS: File		
Database Migrarion Service	Save raw data to S3 landing bucket	arrived	Lambda: Moving file	Save raw data to S3 vault bucket

#### DMS: saving data

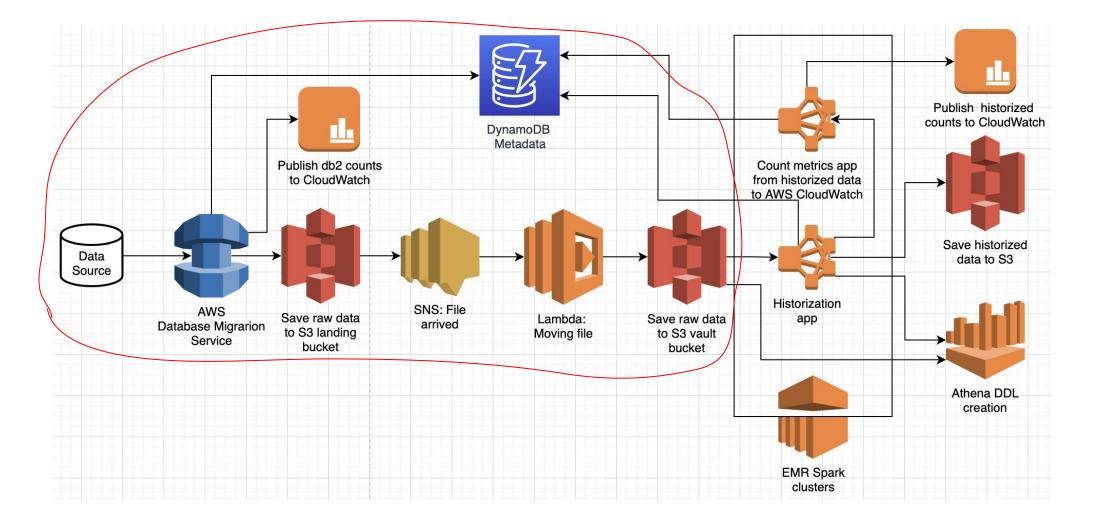




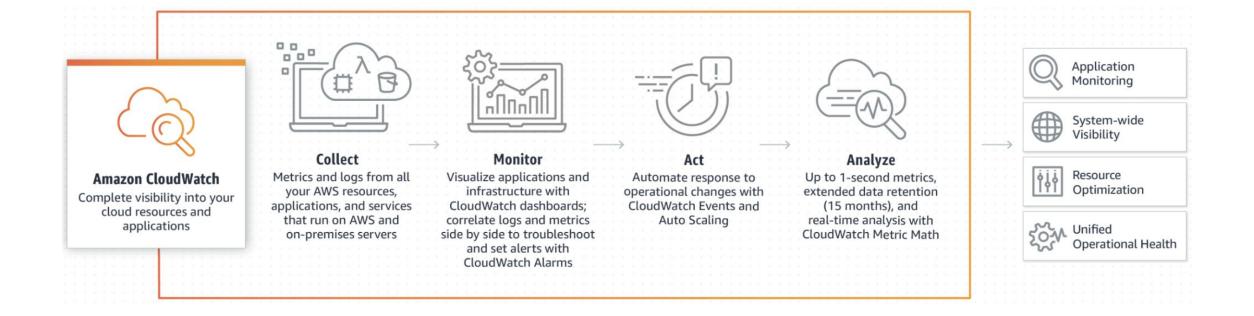


#### First part: DMS

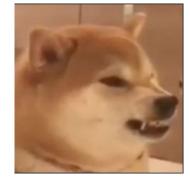














Data Platform product owner







Data Platform product owner

He is happy when he observes metrics

Average db2 count over 30 days with sliding window for 1 week 🥒

1h 3h 12h 1d 3d 1w custom - Number - Actions - 📿 - 🥝

#### 16.6<sup>k</sup> 14.6<sup>k</sup> 1.63<sup>M</sup> 11.9<sup>k</sup> 24<sup>k</sup> 59.4<sup>k</sup>

All metric	Graphed metrics (6/84)	Graph options	Source						
Math exp	ression 🗸 🚱 Dynamic lab	els 🗸 🕜			S	atistic: Avera	ige 🗸 Period:	(multiple)	<ul> <li>Remove all</li> </ul>
	Label		Details			Statistic	Period	Y Axis	Actions
			Historiza	ation • SourceRecordCount • TableName	• Env:	Average	30 Days	< >	-~ 4 4 0
<ul> <li>Image: Image: Ima</li></ul>			Historiza	ation • SourceRecordCount • TableName	e • En	Average	30 Days	< >	-~ 4 4 8
			Historiza	ation • SourceRecordCount • TableName	o_da •	Average	30 Days	< >	-∿ △ @ ©
			Historiza	ation • SourceRecordCount • TableName	• Env:	Average	30 Days	< >	-∿ △ @ ©
			Historiza	ation • SourceRecordCount • TableName	• Env:	Average	30 Days	< >	-∿ △ ৫ ©
			Historiza	ation • SourceRecordCount • TableName	_confi	Average	30 Days	< >	-∿ △ @ ©
			Historiza	ation • SourceRecordCount • TableName	v: det	Average	30 Days	< >	-∿- △ @ ©





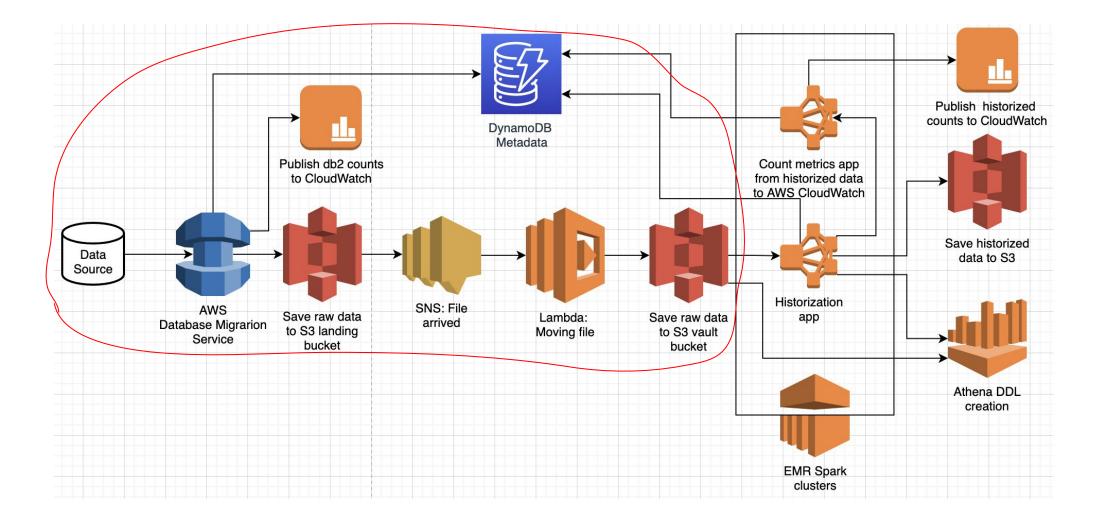


Data Platform product owner

He is happy when he observes metrics

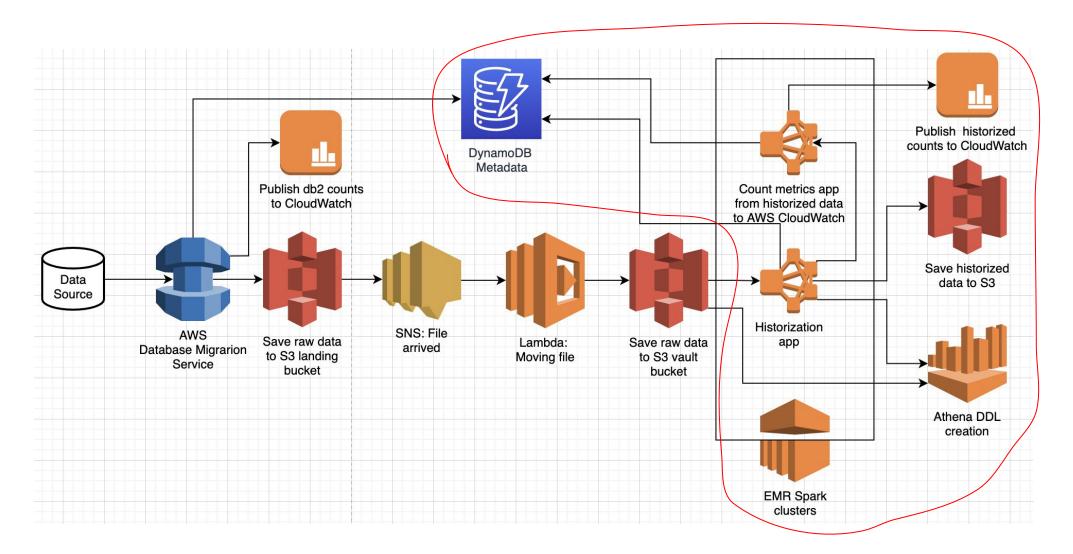
#### First part: DMS





#### Second part: Historization





#### Historization: components





#### Historization: components





# amazon EMR



HRS | Ingestion and historization in the Data Lake

#### Historization: Athena



aws Services V	Q Search for services, features, marketplace products, and docs [Option+S]	∑ 🗘 Aveek22 ▼ Ireland ▼ Support ▼
Athena Query editor Saved queries	History Data sources Workgroup : primary	Settings Tutorial Help What's new
Connect data source AwsDataCatalog  ✓ Database sampledb  ✓ Filter tables and views ✓ Tables (2) Create table ♦ elb_logs	<pre>New query 1  New query 2  +  CREATE EXTERNAL TABLE IF NOT EXISTS sampledb.superstore (</pre>	
You have not created any views. To create a view, run a query and click "Create view from query"	Run query Save as   Create ~ (Run time: 0.27 seconds, Data scanned: 0 KB) Use Ctrl + Enter to run query, Ctrl + Space to autocomplete *** Results Query successful.	Format query     Clear       Athena engine version 1     Release versions

#### Historization: components



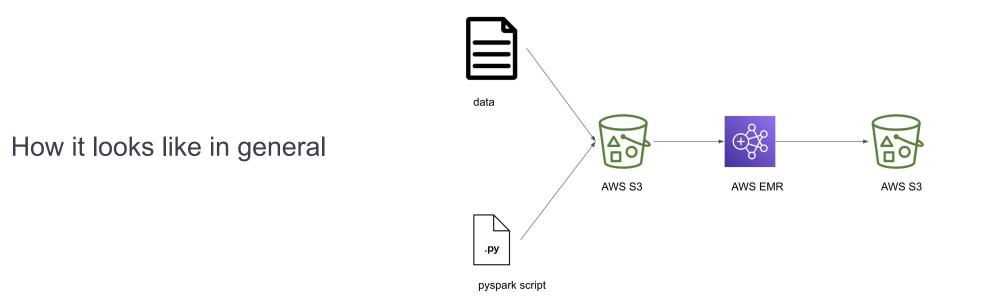






## Spark jobs on EMR cluster





#### Historization: script input params

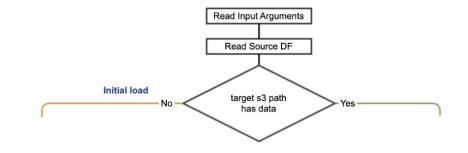


#### Input parameters

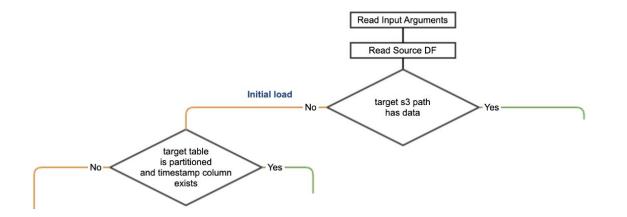
Name	Required	Туре	Description
source_s3_path	True	String	Source path for DMS outputs
target_s3_path	True	String	Target path to the Delta Table
load_date	True	String	Load date for source data in format YYYY-MM-DD
primary_key_fields	True	String	List of fields (comma-separated string) used as primary key for source table
timestamp_column_name	True	String	Timestamp column which will be used for historization to define valid_from and valid_to fields, default 'CTS'
is_target_partitioned	False	Bool	True if target table is partitioned, default is 'True'
is_athena_table_required	False	Bool	True if you want to create/update Athena table and you're sure that table doesn't contain PII data, default is 'False'

1 {	
	"SchemaName": "schema_name",
3	"TableName": "table_name",
4	"Source": "db2",
5	"Target": "s3",
6	"StartValue": "1970-01-01 00:00:00.000000",
	"EndValue": "2021-05-11 00:00:00.000000",
8	<pre>"PrimaryKeyFields": "ID_VALUE",</pre>
9	"TimestampColumnName": "",
0	"OutputFormat": "parquet",
	"BrokenDateTimeColumns": "",
	"DecimalWithNullColumns": "",
3	"IsAthenaTableRequired": true,
4	"IsTargetPartitioned": false
5 }	

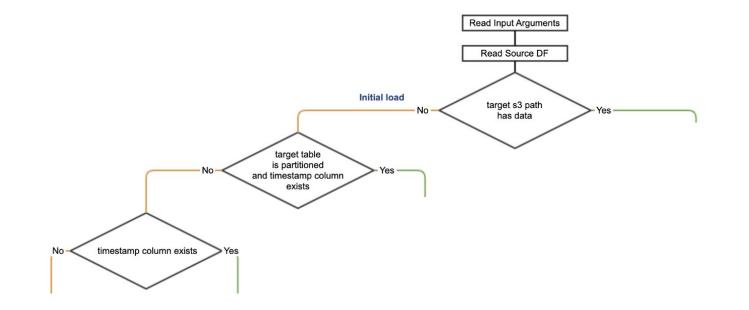




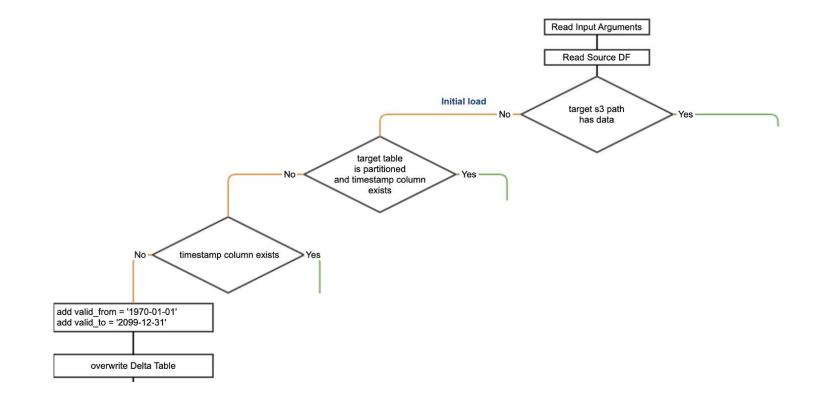




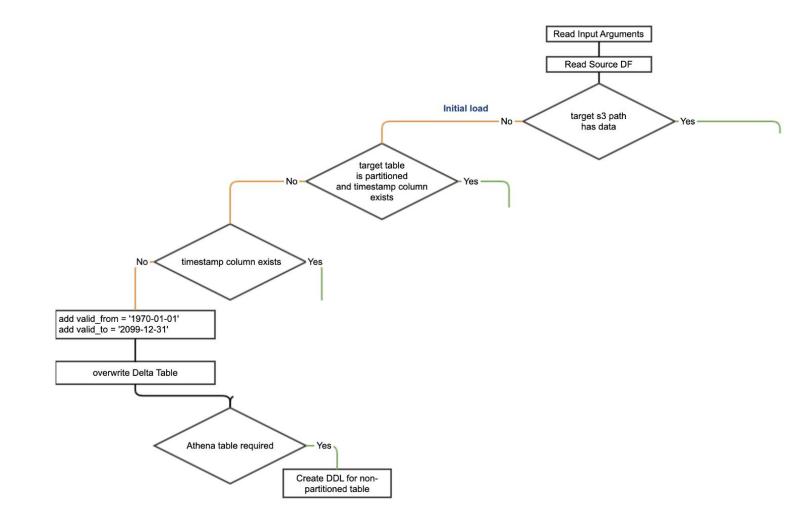




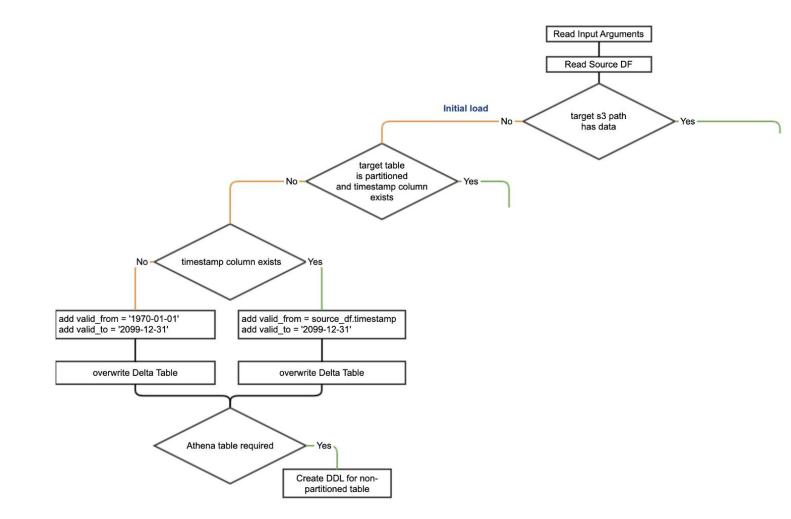




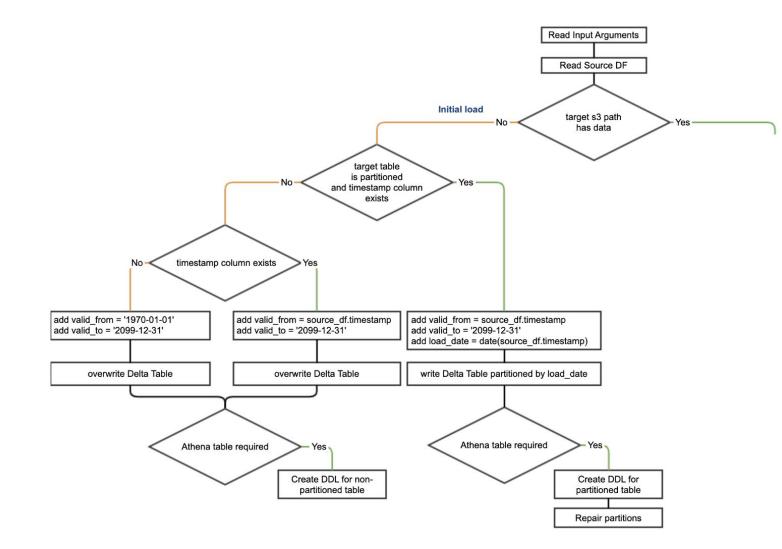




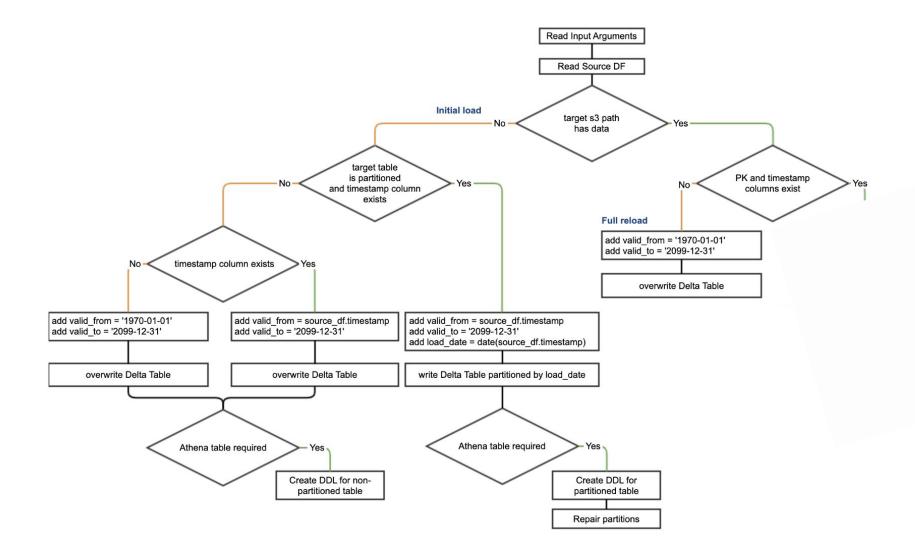




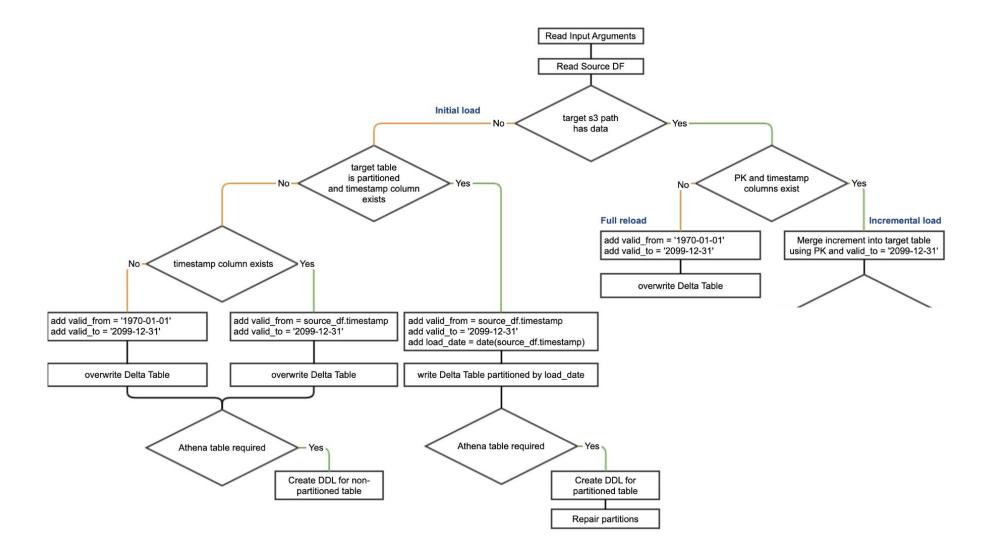




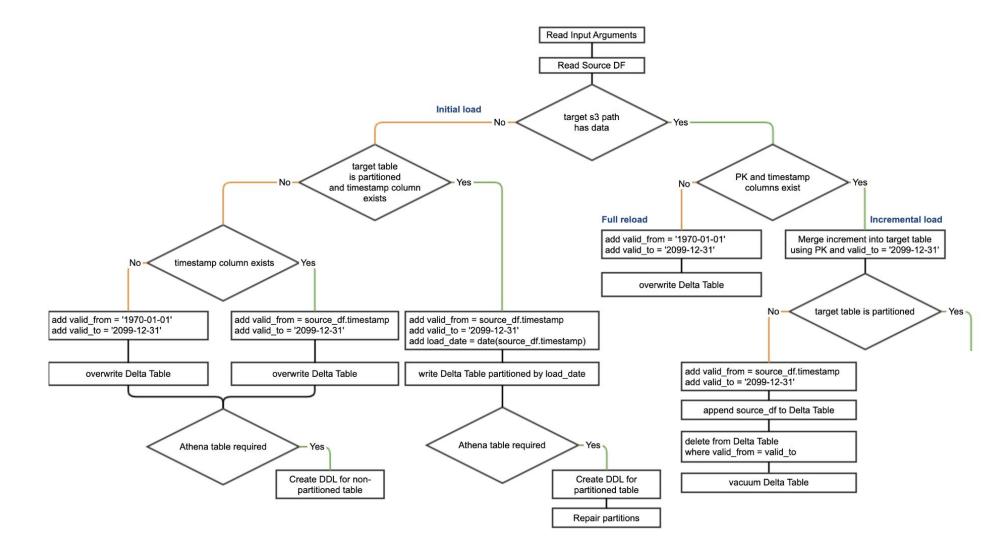




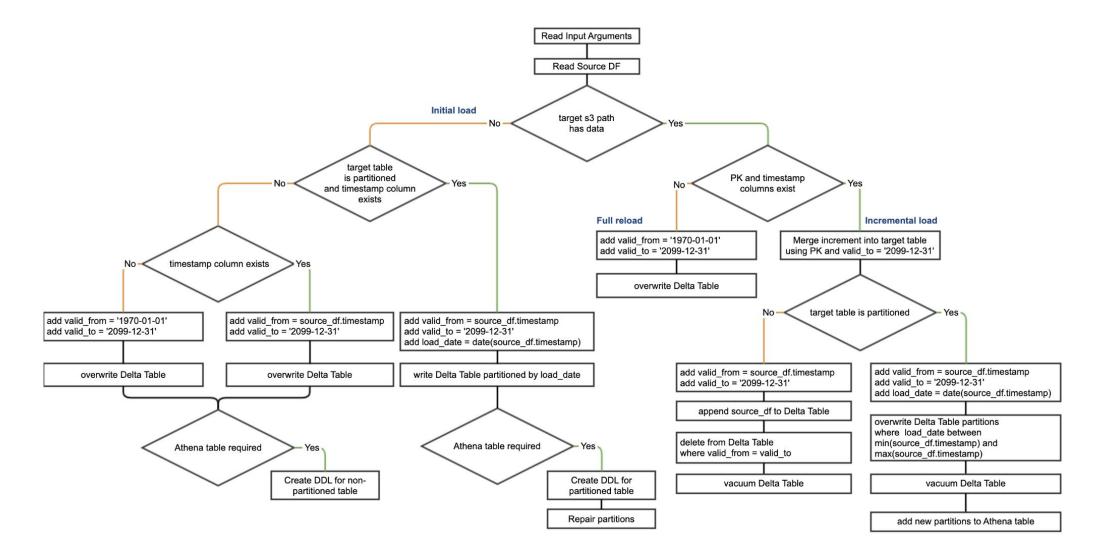












#### Workflow management platform



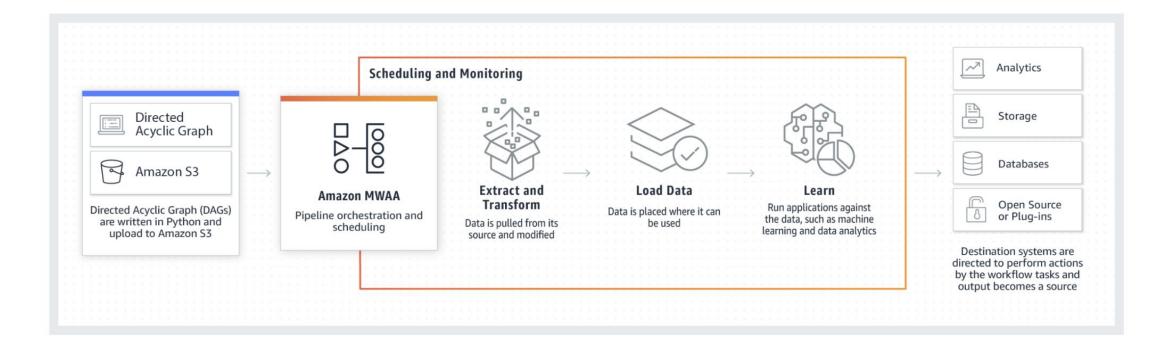
The workflow is orchestrated by MWAA



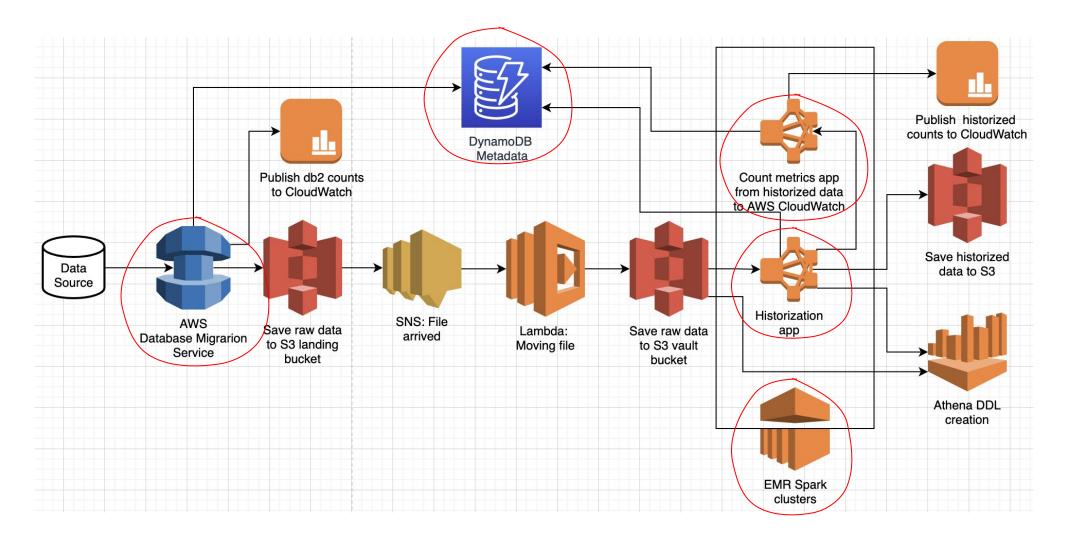
#### Workflow management platform



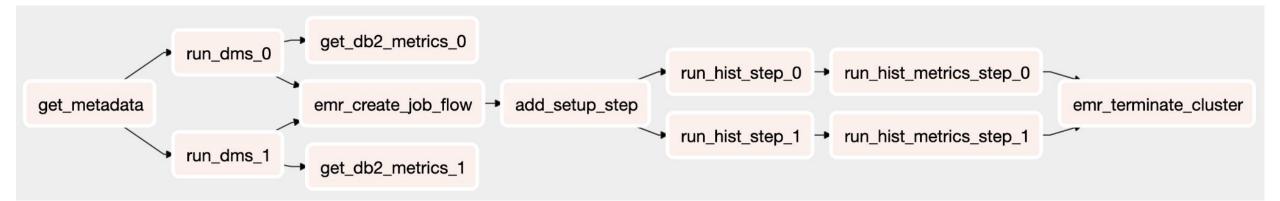
#### The workflow is orchestrated by MWAA



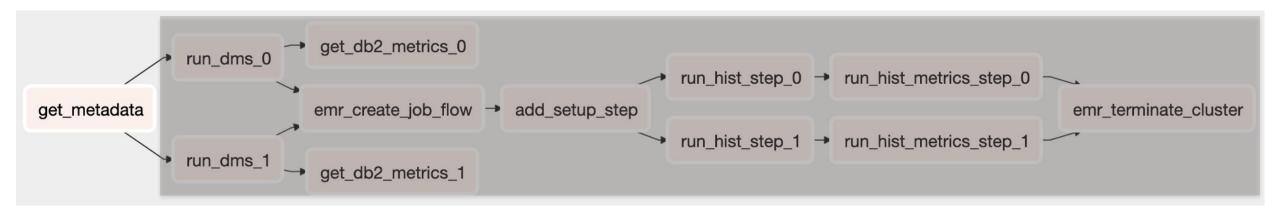




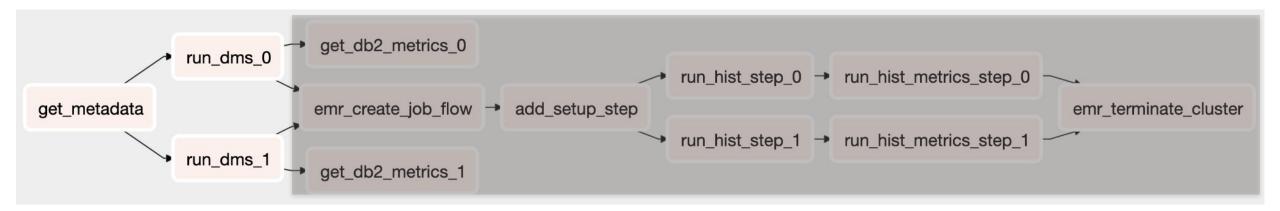




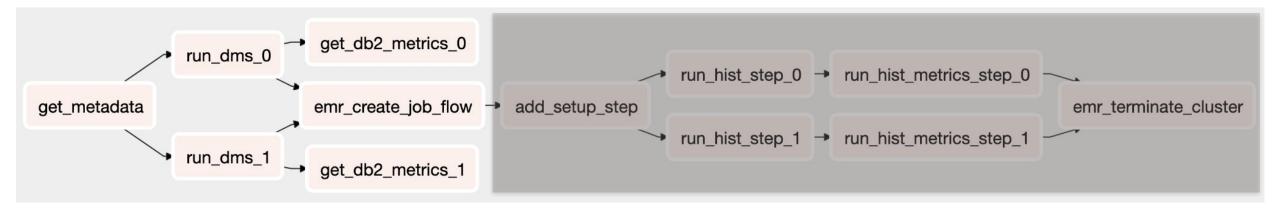




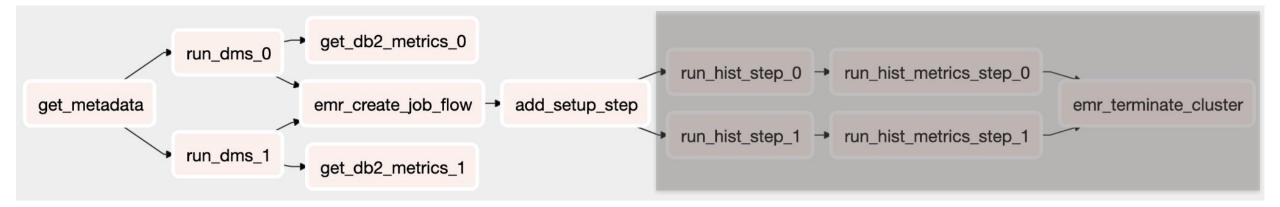




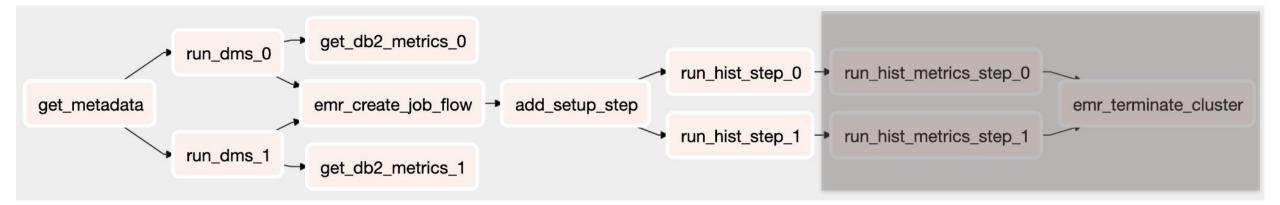




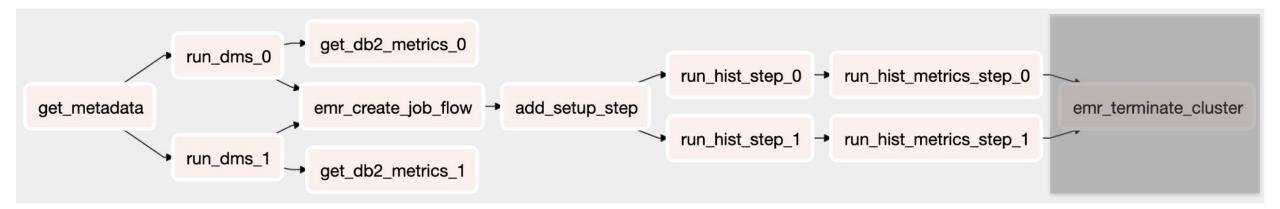




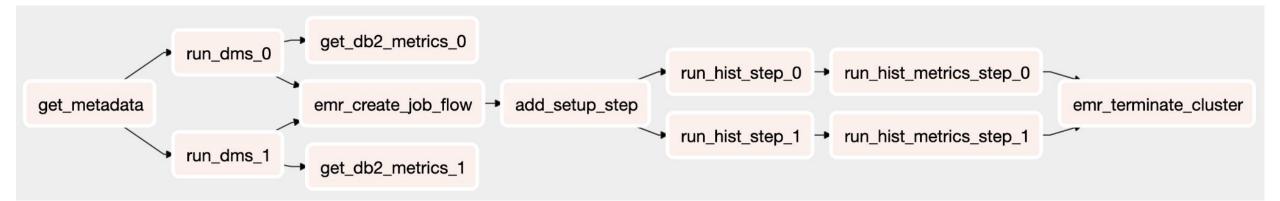




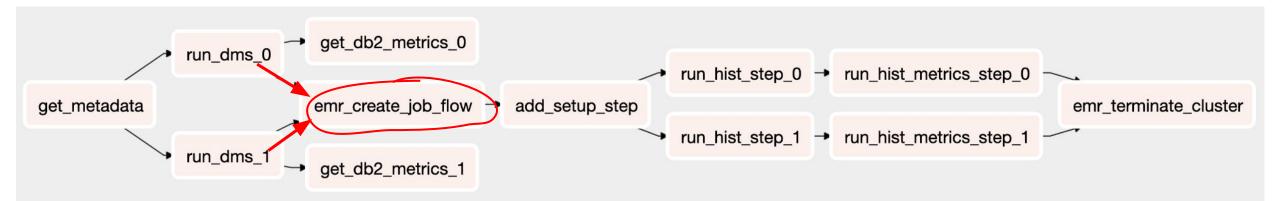






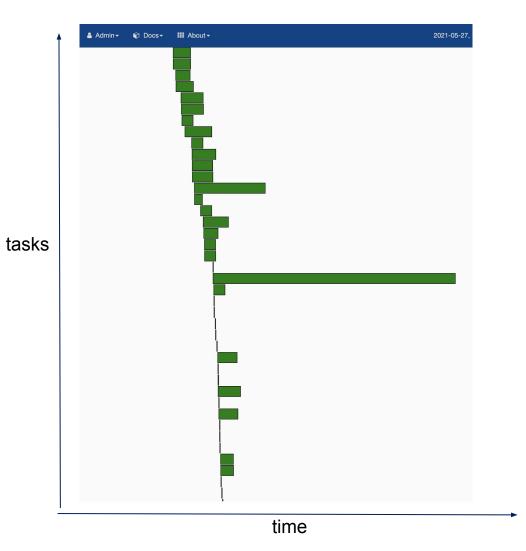






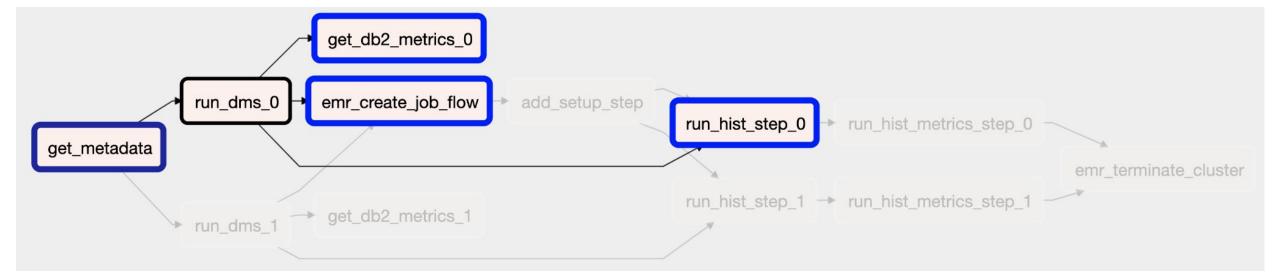
### Dag first iteration: Gantt diagramm





#### Airflow: Dag second iteration





### Airflow: Monitoring and alerting





# Airflow: Monitoring



```
1 # Airflow DAG Config
2 DEFAULT_ARGS = {
3    'email': ['airflow@example.com'],
4    'email_on_failure': True,
5    'email_on_retry': False,
6    'on_failure_callback': send_zoom_failure_message,
7    'retries': 3,
8    'retry_delay': timedelta(minutes=3)
9 }
```

# Airflow: Monitoring



```
1 # Airflow DAG Config
2 DEFAULT_ARGS = {
3    'email': ['airflow@example.com'],
4    'email_on_failure': True,
5    'email_on_retry': False,
6    'on_failure_callback': send_zoom_failure_message,
7    'retries': 3,
8    'retry_delay': timedelta(minutes=3)
9 }
```

# Airflow: Monitoring



```
1 # Airflow DAG Config
2 DEFAULT_ARGS = {
3    'email': ['airflow@example.com'],
4    'email_on_failure': True,
5    'email_on_retry': False,
6    'on_failure_callback': send_zoom_failure_message,
7    'retries': 3,
8    'retry_delay': timedelta(minutes=3)
9 }
```

#### Airflow: points to improve



Cluster:

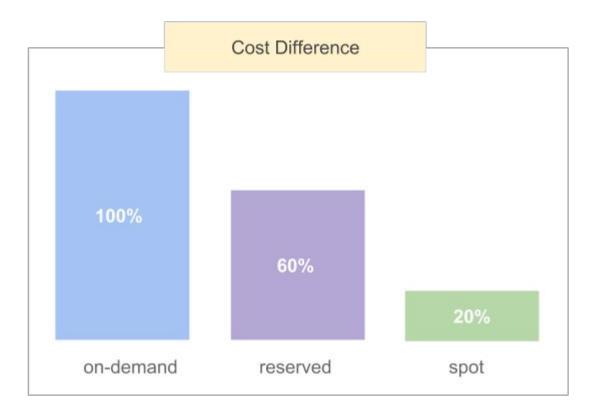
Terminated with errors Master node was terminated due to not enough capacity in the Spot Instance pool

## EMR EC2 Spot Instances



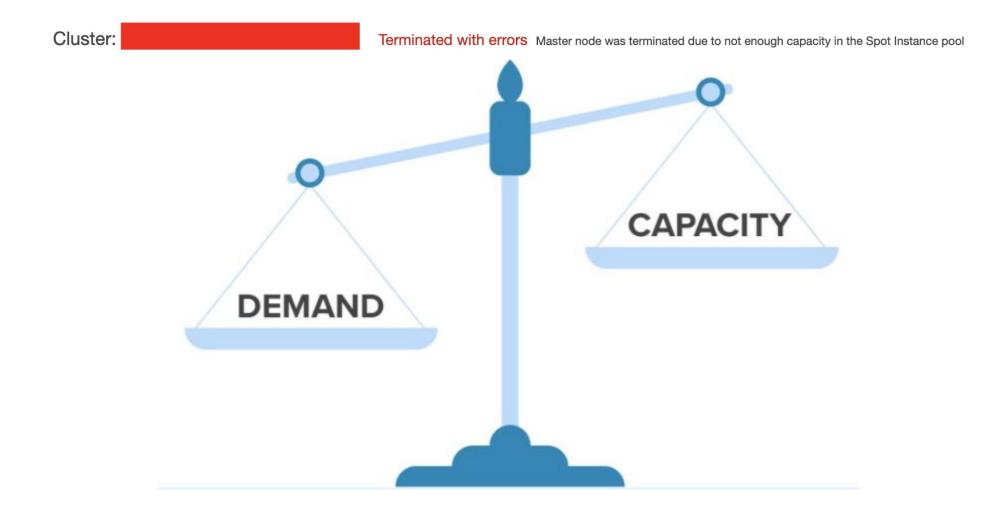
Name your own price for EC2 Compute

- A market where price of compute changes based upon Supply and Demand
- When Bid Price exceeds Spot Market Price, instance is launched
- Instance is terminated (with 2 minute warning) if market price exceeds bid price
- Unused On-Demand Instances



#### Airflow: points to improve

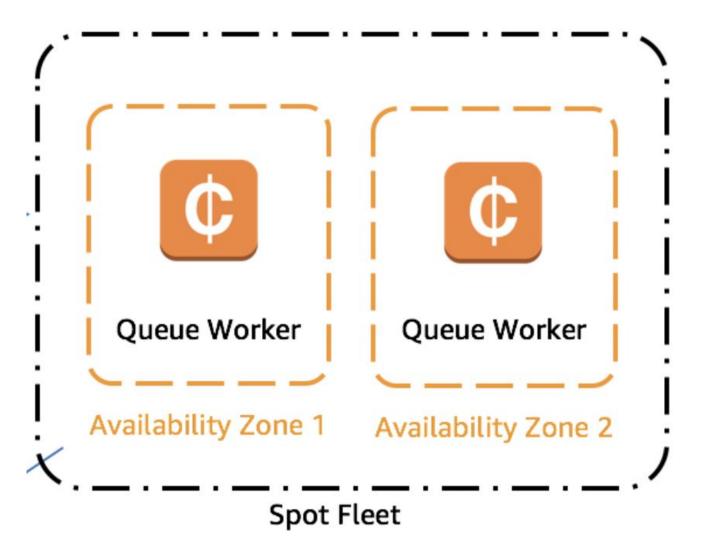




#### Airflow: ec2 fleet



- The instance fleets configuration for a cluster offers the widest variety of provisioning options for EC2 instances
- With instance fleets, you specify target capacities for On-Demand Instances and Spot Instances within each fleet.



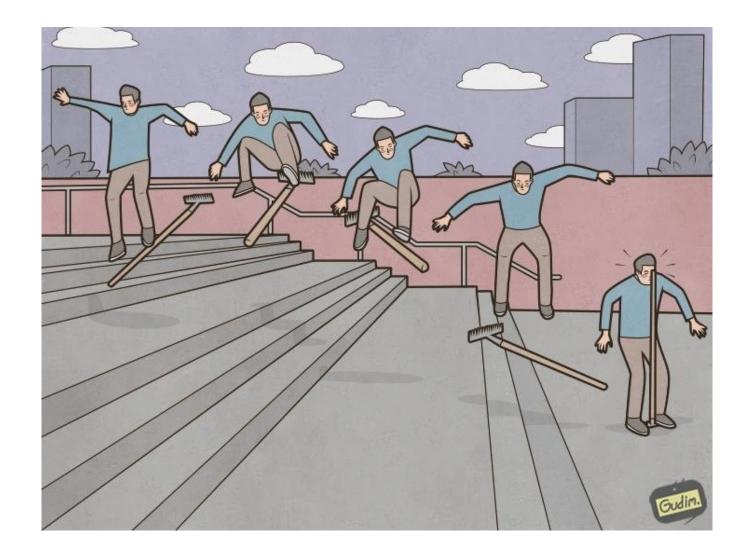
# That's it?





That's it?









• time\_millis

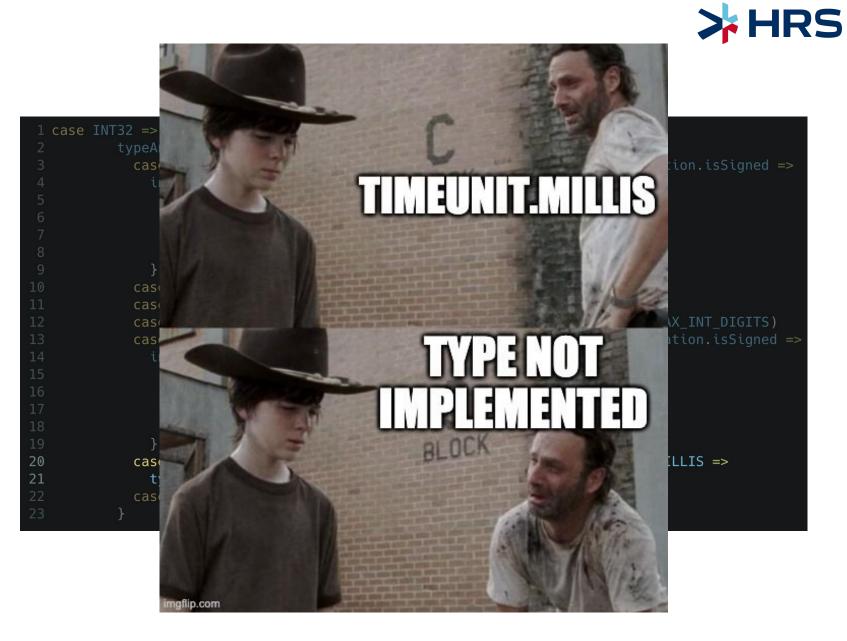


#### Spark ParquetSchemaConverter.scala

1 case INT	32 =>
	typeAnnotation match {
	<pre>case intTypeAnnotation: IntLogicalTypeAnnotation if intTypeAnnotation.isSigned =&gt;</pre>
	<pre>intTypeAnnotation.getBitWidth match {</pre>
	case 8 => ByteType
6	<pre>case 16 =&gt; ShortType</pre>
	<pre>case 32 =&gt; IntegerType</pre>
8	<pre>case _ =&gt; illegalType()</pre>
9	}
	case null => IntegerType
	<pre>case _: DateLogicalTypeAnnotation =&gt; DateType</pre>
	<pre>case _: DecimalLogicalTypeAnnotation =&gt; makeDecimalType(Decimal.MAX_INT_DIGITS)</pre>
	<pre>case intTypeAnnotation: IntLogicalTypeAnnotation if !intTypeAnnotation.isSigned =&gt;</pre>
	<pre>intTypeAnnotation.getBitWidth match {</pre>
	<pre>case 8 =&gt; ShortType</pre>
16	<pre>case 16 =&gt; IntegerType</pre>
	case 32 => LongType
18	<pre>case _ =&gt; illegalType()</pre>
19	}
20	<pre>case t: TimestampLogicalTypeAnnotation if t.getUnit == TimeUnit.MILLIS =&gt;</pre>
21	<pre>typeNotImplemented()</pre>
	<pre>case _ =&gt; illegalType()</pre>
	}

• time\_millis

• time\_millis



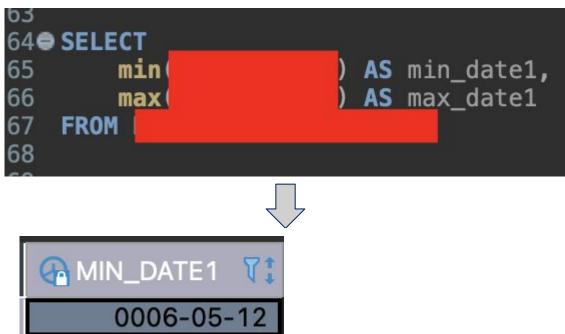


63 64**0** SELECT 65 min() AS min\_date1, 66 max() AS max\_date1 67 FROM 68

- time\_millis
- inadequate dates

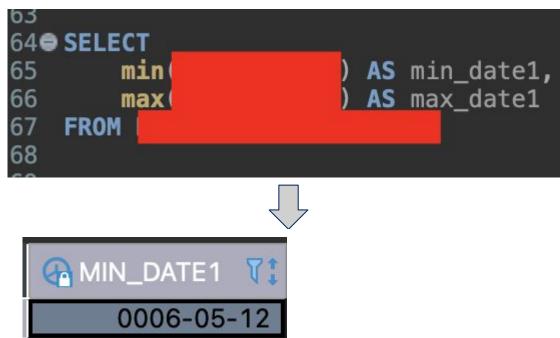
**HRS** 

- time\_millis
- inadequate dates



>> HRS

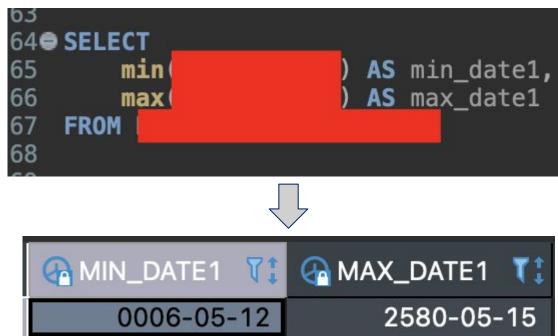
- time\_millis
- inadequate dates





**HRS** 

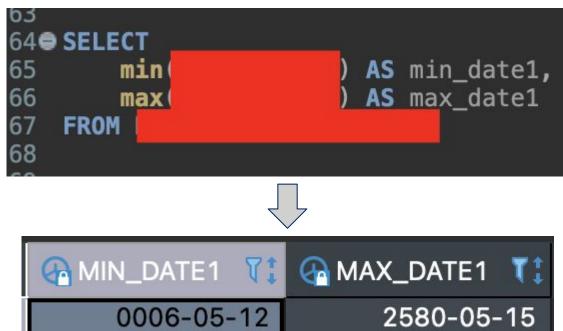
- time\_millis
- inadequate dates





**HRS** 

- time\_millis
- inadequate dates



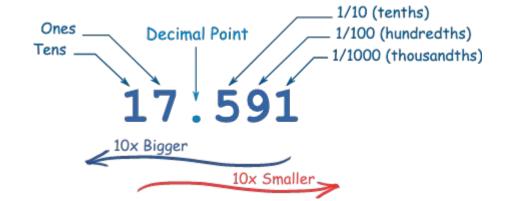


Data type issues

time\_millis

 ${ \bullet }$ 

- inadequate dates
  - decimal





# Data type issues: solution



# Data type issues: solution





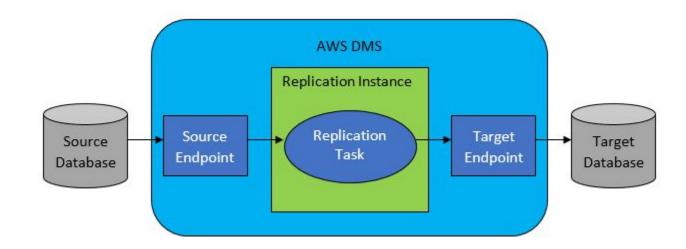
# DMS: Source Endpoint



	DataSourceEndpoint:
2	Type: AWS::DMS::Endpoint
3	Properties:
4	EndpointIdentifier: prod-data-source-endpoint
5	EngineName: data_source
6	EndpointType: source
7	<pre>Username: '{{resolve:secretsmanager:service:SecretString:username}}'</pre>
8	<pre>Password: '{{resolve:secretsmanager:service:SecretString:password}}'</pre>
9	<pre>ServerName: '{{resolve:secretsmanager:service:SecretString:host}}'</pre>
10	<pre>Port: '{{resolve:secretsmanager:service:SecretString:port}}'</pre>
	<pre>DatabaseName: '{{resolve:secretsmanager:service:SecretString:database}}'</pre>
12	<pre>ExtraConnectionAttributes: 'executeTimeout=3600;'</pre>

#### DMS: ReplicationTask fun part





## DMS: ReplicationTask fun part



No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

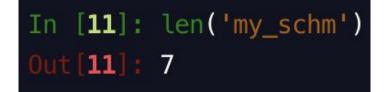


No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

DataSource schema: my\_schm



No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern



DataSource schema: my\_schm



No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

# In [11]: len('my\_schm') Out[11]: 7

DataSource schema: my\_schm



No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

# In [11]: len('my\_schm') Out[11]: 7

DataSource schema: my\_schm





No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

## In [11]: len('my\_schm') Out[11]: 7

DataSource schema: my\_schm

We are in process of updating the documentation with this limitation.

Being said that there is a workaround to select one specific table by changing "rule-action" from "include" to "explicit"





No tables were found at task initialization. Either the selected table(s) no longer exist or no match was found for the table selection pattern

## In [11]: len('my\_schm') Out[11]: 7

DataSource schema: my\_schm

We are in process of updating the documentation with this limitation.

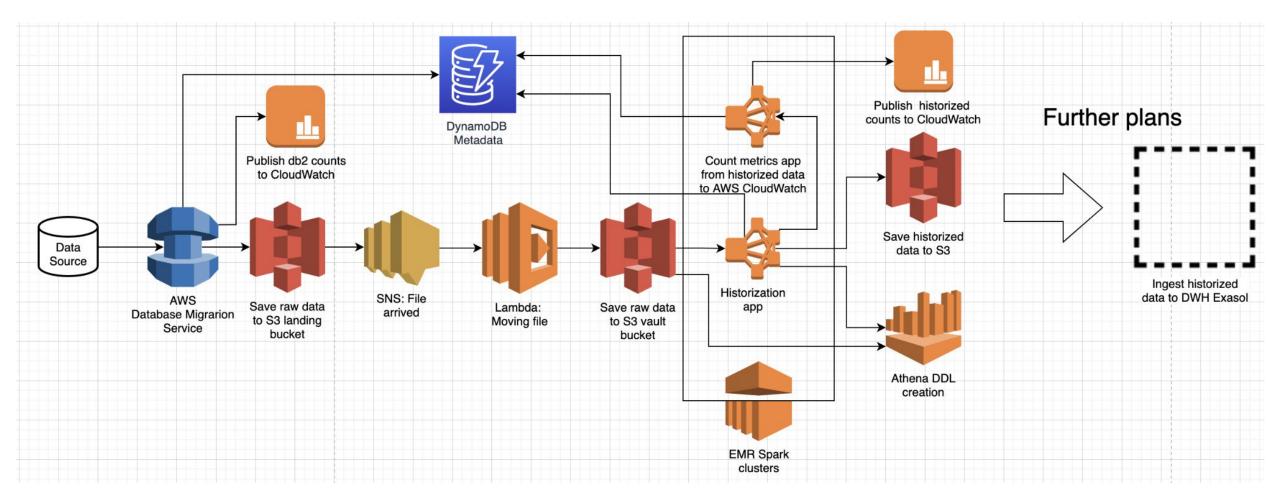
Being said that there is a workaround to select one specific table by changing "rule-action" from "include" to "explicit"

rule-	include, exclude, explicit	A value that includes or excludes the object or objects selected by the rule. If explicit is specified, you
action		can select and include only one object that corresponds to an explicitly specified table and schema.



#### Further work













• We have built ETL process for full and incremental load data from Data Source to Data Lake





- We have built ETL process for full and incremental load data from Data Source to Data Lake
- We have automated and auto scaled process





- We have built ETL process for full and incremental load data from Data Source to Data Lake
- We have automated and auto scaled process
- About 70 tables are running day by day





- We have built ETL process for full and incremental load data from Data Source to Data Lake
- We have automated and auto scaled process
- About 70 tables are running day by day
- We have just one entry point for adding new tables to the process all other steps are catching up and reusing this metadata





- We have built ETL process for full and incremental load data from Data Source to Data Lake
- We have automated and auto scaled process
- About 70 tables are running day by day
- We have just one entry point for adding new tables to the process all other steps are catching up and reusing this metadata
- We enabled process monitoring and alarming on data and process issues





- We have built ETL process for full and incremental load data from Data Source to Data Lake
- We have automated and auto scaled process
- About 70 tables are running day by day
- We have just one entry point for adding new tables to the process all other steps are catching up and reusing this metadata
- We enabled process monitoring and alarming on data and process issues
- Our data stored in effective way and easy to access with different tools

## Learning points to you





## Q&A: Your voice matters







Link to the anonymous survey

### Metadata in DynamoDB



Partition key: **TableID** is the following composition **<Source>-<Target>-<SchemaName>-<TableName>** 

#### Metadata in DynamoDB



Partition key: **TableID** is the following composition **<Source>-<Target>-<SchemaName>-<TableName>** 

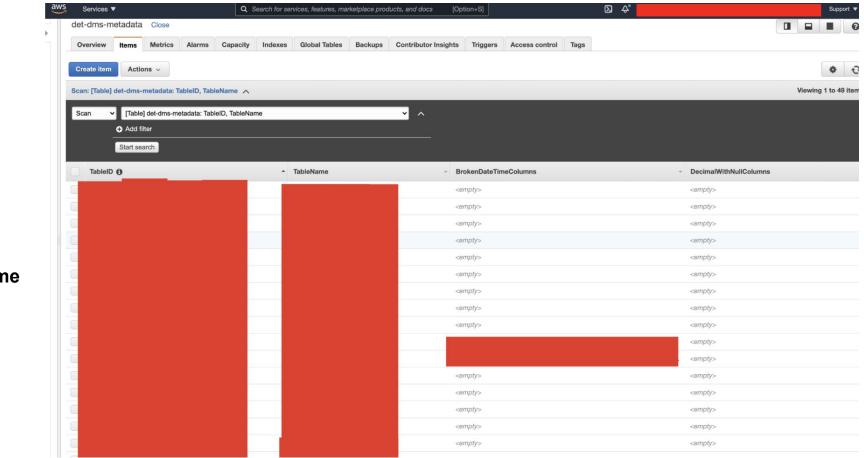
Sort key: TableName

### Metadata in DynamoDB



Partition key:

TableID is the following composition <Source>-<Target>-<SchemaName>-<TableName>



Sort key: TableName