

Design - Commercetools Import Containers Reporting

- [Objective](#)
- [Import Containers](#)
- [NGC Catalog Import Jobs](#)
- [High Level Solution](#)
 - [Technical Notes](#)
 - [Import Summaries](#)
 - [Import Operations](#)
 - [Important Queries to generate the report](#)
- [Container Naming Strategy for PCM and CM Catalog](#)
 - [PCM](#)
 - [Convention](#)
 - [Example](#)
 - [CM Full Load - Create 46 containers](#)
 - [Convention](#)
 - [Example](#)
 - [CM Incremental Load](#)
 - [Convention](#)
 - [Example](#)
- [Abbreviations](#)
- [References](#)

Objective

In NGC, all the data for Catalog such as categories, products, SKUs (Called product variants in CT), their relationships and prices are getting ingested using the Commercetools(CT) asynchronous Import APIs. Because of API Imports async nature, its not possible to know what happened with that transaction (Operation). The only id that gets return is the operation Id for that transaction. There are 3 things that could happen with the operation as stated below

- **Successful** - Import Operation is successful, for e.g. resource such as Category has been successfully ingested using Import Operation
- **Unresolved** - Resource such as Product has a dependency on other resource such as Category which is yet to be ingested, and Product Import is waiting to get it resolved
- **Failed** - Some mandatory attribute is not present on the resource

NGC already has aggregator and Ingestor services to ingest data from PCM and CM using the CT's Import API Operations and operation ids are being returned. A report needs to be generated in Splunk to see the status of those operations so that data can be corrected if needed in the source system or fix the bug in the existing Aggregator and Ingestor services

Import Containers

An Import Container is the entry point of the Import API requests and serves as a data container for [asynchronous](#) API calls. Every create, update, or patch request sent to the Import API is first accumulated in an [Import Container](#) and then asynchronously imported into the platform. In NGC, different Import containers are being used to ingest the Catalog data from PCM and CM sources.

Import Containers can be used to organize import requests. For example, in NGC, the data is imported from 2 sources i.e PCM and Catalog Master, and we're creating Import Containers for each source. Creating different containers for PCM and catalog master enables NGC to monitor the progress of import by the source by passing Import Containers as a path parameter.

When NGC performs full data imports, it is useful to create a new Import Container every time to distinguish imports performed on different occasions.

⚠ An [Import Operation](#) is automatically deleted in **48** hours after it is created. As there are no limits to how many containers can be created, we can leave the created containers as is for now, and then delete them after X hours by running a job at scheduled time

NGC Catalog Import Jobs

In NGC, data is being ingested in CT from 2 systems

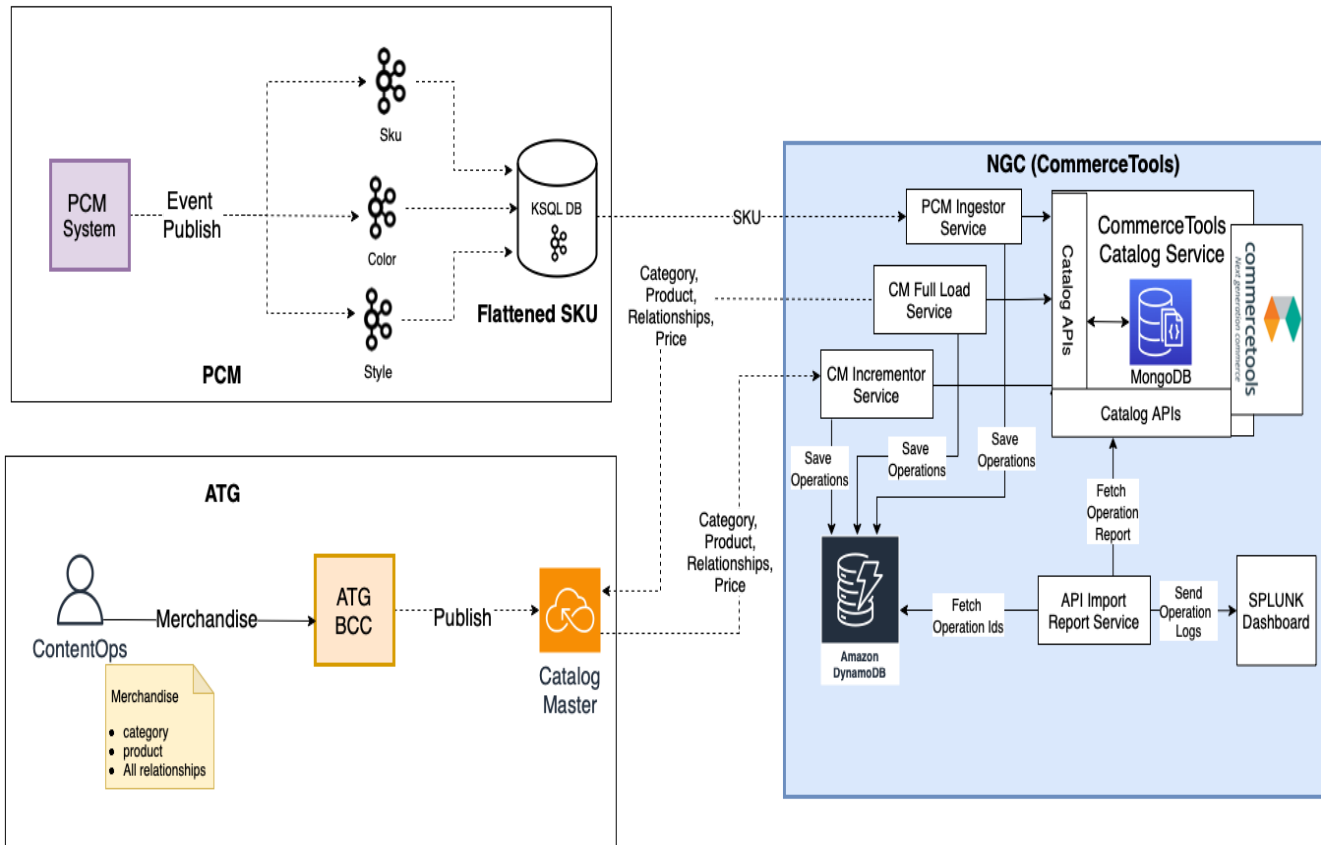
- From PCM for all the SKUs, there is single PCM Ingestor Service for full and incremental load (Service deployed on EKS)

[-] Component needs to be created to insert the Container's name, Operation Ids, and resource identifiers in Dynamo DB from this service

- From Catalog Master (CM) for all the categories, products, their relationships, SKUs and prices, there are 3 services. 2 services for Full load and 1 for Incremental load
 - CM Full Load Aggregator (Serverless Service) - **Nothing to be done in this service as it has no interaction with the CT**
 - CM Full Load Ingestor (Service deployed on EKS)
 - CM Incremental Load Ingestor (Serverless Service)

[-] Component needs to be created to insert the Container's name, Operation Ids, and resource identifiers in Dynamo DB from CM Full Load and Incremental Ingestor services

High Level Solution



Technical Notes

1. Use Dynamo DB from Catalog Ingestor Service to save operation ids, and container ids and any other useful data such as SKU, product, and category identifiers.
2. Write a separate ngc-catalog-report-service serverless that will read operation and container ids from the Dynamo DB, make an API call, and log the information
3. Create Dashboards in Splunk, and raise alerts for anything that needs fix
4. Put a TTL for every record in Dynamo DB
5. Persisting in Dynamo DB from the PCM and CM Ingestor should be asynchronous and should not impact the performance of the APIs in these services
6. Use the Scheduler for retries may be 3 attempt exponentially
7. Maintain a flag in Dynamo DB for the Operations so that we don't query that again for the operations that have been successfully ingested
8. Persist in Dynamo DB
 - a. Type of the resources such as Product, SKU, Price, etc.
 - b. Retried attempts if any for querying the operations
 - c. error that has happened in CT for operations
9. ~~To analyze, if report can be created using CloudWatch + Dynamo DB – Nice to have for NGC MVP~~

Import Summaries

An Import Summary describes the status of an [Import Container](#) by the number of resources in each [Processing State](#). It can be used to monitor the progress of import per [Import Container](#).

To monitor the status of an Import Container more in detail, use [Query ImportOperation](#). To monitor the status of individual resource, use [Get ImportOperation](#).

Import Operations


An [Import Operation](#) is a status of the import of an individual resource. Import Operations can be retrieved by ID or can be queried within a specific [Import Container](#). Both Import Operations and [Import Summary](#) can be used to monitor import progress. The former focuses on the status of each import resource. The latter reports the status of an Import Container by the number of resources in each [Processing State](#).


An [Import Operation](#) is automatically deleted in **48** hours after it is created.


Important Queries to generate the report


 [Import Operations](#)

 [Import Operations](#)

 [Import Summaries](#)

 SDK Equivalent for above queries to be used

 Use the postman collection from this link to play with the APIs during the development [commercetools-postman-collection/import](#)
[at master · commercetools/commercetools-postman-collection](#)

 Responses from Hetvi from CT:

"Regd. your queries about import api operations API. Please find below:

i) Ability to query import containers for multiple statuses - This is currently not possible, You have to make one API call per status when searching operations for a particular status

ii) Ability to query import containers by date range or container name -

-You can use Get Import-Container to search by container-key

-Search by date range is not available however, you can use sort functionality to sort the results by latest to get most recent container operations

iii) Ability to pass multiple operation ids to fetch records within a container - This is not available currently

Fetching operations result 500 operations in a result, however you can implement pagination

iv) Can 48 hours window of import operations be configurable - No, this value is a product feature and cannot be configured."

Container Naming Strategy for PCM and CM Catalog


In order to create a report for API Import Operations its imperative to understand the API Import container's strategy.

PCM

1. pcm-product_0_date
2. pcm-product_1_date
3. pcm-product_2_date
4. pcm-product_3_date

Convention

pcm-product_X_DD-MM-YY in PST timezone

 Green is static key and red needs to be dynamically generated during runtime, and appended with the static key

Example

First container #0 created for PCM ingestion on Mar 12, 2022 in PST


- pcm-product_0_12-03-22

CM Full Load - Create 46 containers

1. cm-fullload-category_CSVFileNameFromS3 (1 container)
2. cm-fullload-product-variant-patch_CSVFileNameFromS3 (15 Containers per product CSV file)
3. cm-fullload-product_CSVFileNameFromS3 (15 Containers per product CSV file)
4. cm-fullload-price_CSVFileNameFromS3 (15 Containers per product CSV file)

Convention

cm-fullload-category_CSVFileNameFromS3

 Green is static key and red needs to be dynamically generated during runtime, and appended with the static key

Example

First container created for CM full load ingestion for categories on Mar 12, 2022 at 18 hour, 22 minute and 35 seconds in PST

cm-fullload-category_CSVFileNameFromS3


CM Incremental Load

1. cm-incremental-category_date
2. cm-incremental-product_date

3. cm-incremental-price_date

Convention

cm-incremental-category_DD-MM-YY in PST timezone

 Green is static key and red needs to be dynamically generated during runtime, and appended with the static key

Example

First container created for CM incremental load ingestion for categories on Mar 12, 2022 in PST

cm-fullload-category_12-03-22

Abbreviations

- CM - Catalog Master
- CT - Commercetools
- PCM - Product Catalog Model

References

 [Import Containers](#)