



Framework d'Accélération des Données SAP en utilisant Snowflake et Technologies Associées

Date de présentation : 24/06/2024



M'hamed Issam ED-DAOU

Stagiaire Data Engineer
Cloud chez VISEO



SOMMAIRE

- 01. Introduction
- 02. Architecture du Framework
- 03. SAP RAW LANDING
- 04. LANDING TO BRONZE
- 05. BRONZE TO SILVER
- 06. SILVER TO GOLD
- 07. Conclusion et Prochaines étapes

1.

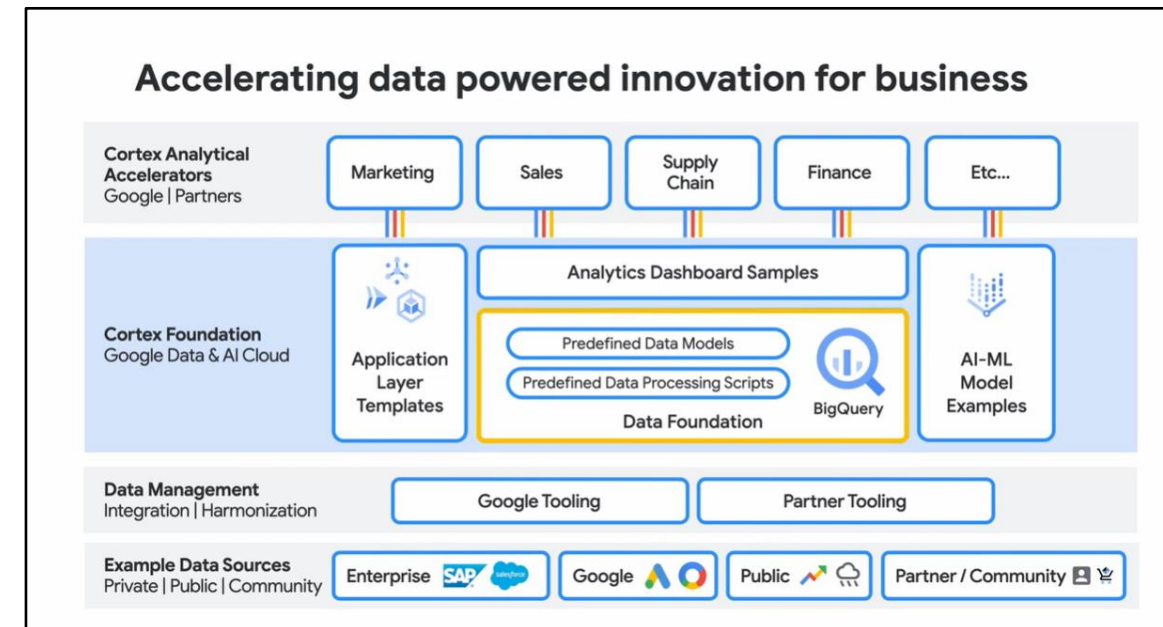
Introduction

INTRODUCTION

Objectifs et Présentation Fonctionnelle

GOOGLE CLOUD CORTEX ?

- Framework Open-Source
- Sources de Données Multiples (SAP, Salesforce, Google Ads...)
- Basé sur les services GCP (BigQuery, Cloud Composer...)
- Solutions Innovatives



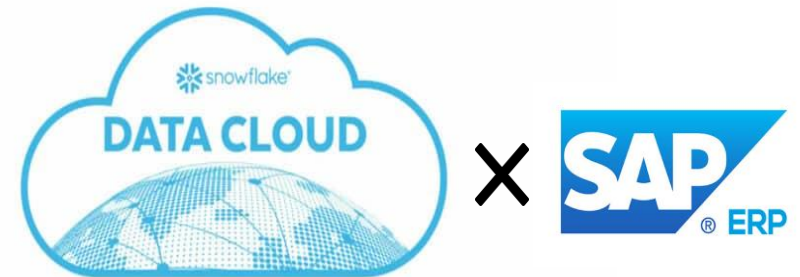
Src : <https://cloud.google.com/solutions/cortex>

INTRODUCTION

Objectifs et Présentation Fonctionnelle

OBJECTIFS ?

- **Concevoir et Développer** un Fonctionnement Similaire du Cortex avec **Snowflake et Technologies Associées**
- **Proof of Concept (POC):** Passer des fichiers **CSV** et **JSON** qui présentent la source des données **SAP ECC** vers des **Tables Fonctionnelles** prête pour l'analyse et le reporting **métier**





2.

Architecture du Framework



 **SYSADMIN**
DEV_VISEO

 **DEV_WH**

 **DEV_DB_VISEO**

Medallion Architecture

SAP_LANDING

SAP_BRONZE

SAP_SILVER

SAP_GOLD

Consumption

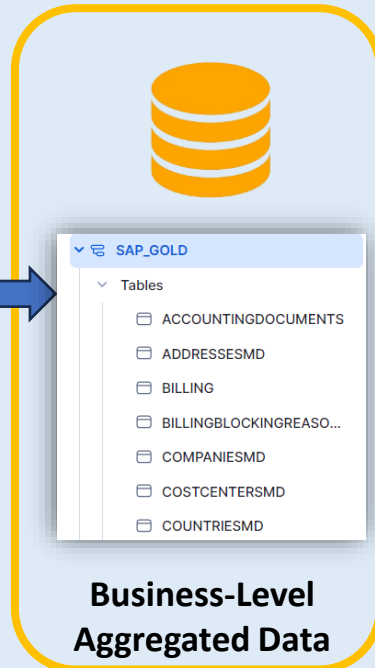
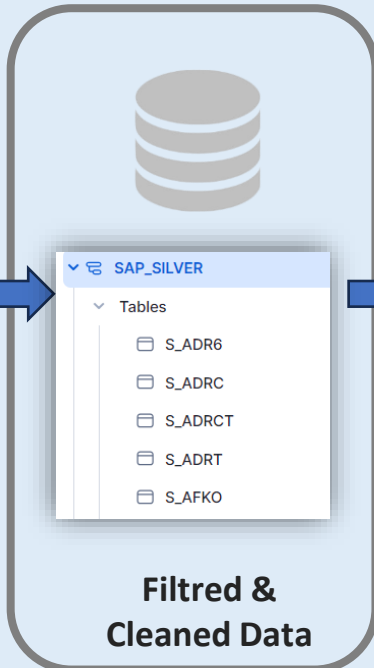
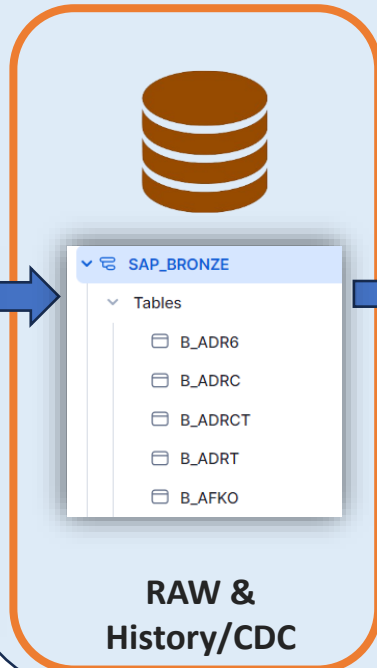
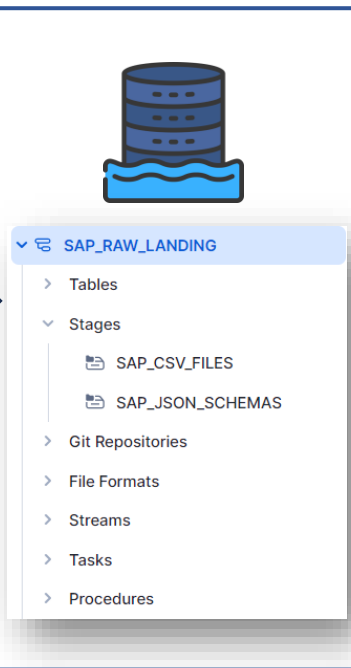
Local File System



SAP Data Files



SAP Tables Schema



Cloning / Views

VendorPerformance

SalesOrders

OderToCash

PUT File
Command

Bulk Loading
Automated

MERGE Query
Automated



Git repo

Create CI/CD pipelines
with GitHub Actions

3.

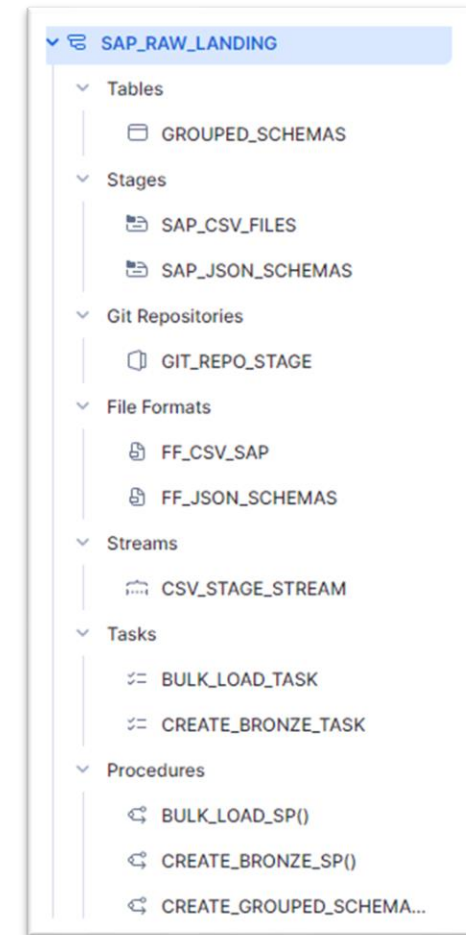
SAP RAW LANDING

SAP RAW LANDING LAYER

Structure du schéma SAP_RAW_LANDING

MAIN INTEGRATIONS

- Ingestion des Fichiers **CSV** et **JSON** dans des Stages ✓
- **Flattening/Applatissement** des fichiers Json et Regroupement en **une seule table** ✓
- Développement des **File Formats** adaptées avec la nature des données ✓
- Développement des Procédure stockées afin de :
 - Transformer les données en Json (semi-structuré) en une seule table (données structurées) ✓
 - Créer de la structure des tables BRONZE (Normalisation, Règles de gestion...) ✓
 - Alimenter les tables BRONZE (Process automatisée sur >> 100 tables !) ✓
- Incorporation de la **Feature** Git Integration ✓
- Développement d'un **Stream** afin de détecter un changement dans les Stages ✓
- Développement des Snowflake **Tasks** ✓



Src : Snowflake


Database : DEV_DB_VISEO

Schema : SAP_RAW_LANDING

SAP RAW LANDING LAYER

Process de passage TO LANDING

```
$ upload_files.sh
1  #!/bin/bash
2
3  # Définir schema de la source et le stage
4  source_path="/mnt/c/Temp/CSV_Files"
5  stage_path="@SAP_CSV_FILES"
6
7  # Boucle pour chaque fichier present
8  for filepath in "$source_path"/*; do
9      # Modifier le nom du dossier
10     filename=$(basename "$filepath" | cut -f 1 -d '.')
11
12     # Construct the target path in the stage
13     target_path="$stage_path/$filename/"
14
15     # Executer la commande PUT afin de charger les fichiers (chacun dans un dossier convenable) dans Snowflake
16     echo "PUT file://$filepath $target_path;"
17     snowsql -q "PUT file://$filepath $target_path;"
18 done
19
```



DEV_DB_VISEO / SAP_RAW_LANDING / SAP_JSON_SCHEMAS

Internal Stage SYSADMIN 4 days ago

Stage Files Stage Details

SAP_JSON_SCHEMAS / JSON_SCHEMAS_V01 (127 Files)

NAME	SIZE	LAST MOD...	
vbfa.schema.json.gz	416.0B	2 hours ago	...
vbak.schema.json.gz	1008.0B	2 hours ago	...
vbep.schema.json.gz	528.0B	2 hours ago	...
vbpa.schema.json.gz	288.0B	2 hours ago	...
vbrk.schema.json.gz	752.0B	2 hours ago	...
t161.schema.json.gz	480.0B	2 hours ago	...
tcurr.schema.json.gz	224.0B	2 hours ago	...
tspa.schema.json.gz	192.0B	2 hours ago	...
tvkot.schema.json.gz	192.0B	2 hours ago	...
t134t.schema.json.gz	192.0B	2 hours ago	...
tj02t.schema.json.gz	192.0B	2 hours ago	...

Stage réservé aux fichiers JSON



DEV_DB_VISEO / SAP_RAW_LANDING / SAP_CSV_FILES

Internal Stage SYSADMIN 4 days ago

Stage Files Stage Details

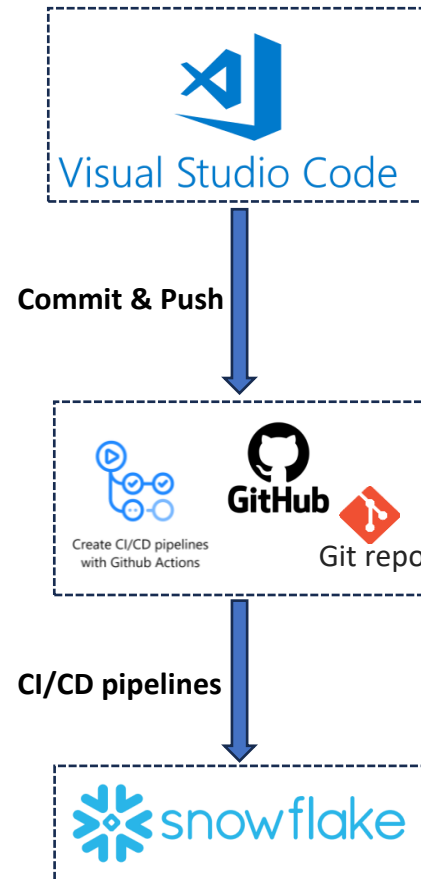
SAP_CSV_FILES (117 Files)

NAME	SIZE	LAST MODIF...	
adr6	384.0B	14 minutes ago	...
adrc	24.4KB	14 minutes ago	...
adrcr	44.0KB	14 minutes ago	...
adrt	121.3KB	14 minutes ago	...
afko	656.0B	14 minutes ago	...
afpo	432.0B	14 minutes ago	...
ankt	16.3KB	14 minutes ago	...
anla	46.9KB	14 minutes ago	...
aufk	656.0B	14 minutes ago	...
bkipf	20.1MB	14 minutes ago	...
bseg	74.6MB	12 minutes ago	...

Stage réservé aux fichiers CSV

SAP RAW LANDING LAYER

Process de déploiement de la couche Landing



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Displays the file structure of the repository. The file `! deploy_sap_raw_landing.yml` is highlighted with a red box. Other files include `! deploy_data_marts.yml`, `! deploy_gold_tables.yml`, `! deploy_gold_udfs.yml`, `! deploy_sap_bronze.yml`, `! setup_snowflake_env.yml`, `dev_poc_sap`, `POC_SAP_BRONZE`, `POC_SAP_GOLD`, `POC_SAP_LANDING`, `DEV_BULK_LOAD_CSV_TO_TABLES.py`, `DEV_BULK_LOAD_SP.sql`, `DEV_CREATE_BRONZE_SP.sql`, `DEV_CREATE_BRONZE_TASK.sql`, `DEV_CREATE_STAGES.sql`, `DEV_CREATE_TABLES_BRONZE.py`, `DEV_FILE_FORMAT_JSON_1.0.0.sql`, `DEV_FILE_FORMATS_CSV_2.0.0.sql`, `DEV_GIT_INTEGRATION.sql`, `DEV_SP_GROUP_SCHEMAS.sql`, `DEV_SP_LOAD_GROUPED_JSON.sql`, `DEV_STREAM_ON_STAGE.sql`, `README.md`, `POC_SAP_SILVER`, `images`, `poc_setup`, and `README.md`.
- WORKSPACE:** Displays the content of the `! deploy_sap_raw_landing.yml` file. The workflow is defined as follows:

```
1 name: Workflow to Deploy SAP_RAW_LANDING Schema Objects
2
3 on:
4   push:
5     branches:
6       - main
7   paths:
8     - '.github/workflows/deploy_sap_raw_landing.yml'
9     - 'dev_poc_sap/POC_SAP_LANDING/**'
10 workflow_dispatch:
11
12 jobs:
13   deploy_sap_raw_landing:
14     runs-on: ubuntu-latest
15
16     env:
17       SNOWSQL_PWD: ${ secrets.SNOWFLAKE_PASSWORD }
18       SNOWSQL_ACCOUNT: ${ secrets.SNOWFLAKE_ACCOUNT }
19       SNOWSQL_USER: ${ secrets.SNOWFLAKE_USERNAME }
20       SNOWSQL_DATABASE: ${ secrets.SNOWFLAKE_DATABASE }
21       SNOWFLAKE_SCHEMA: ${ secrets.SNOWFLAKE_SCHEMA }
22       SNOWSQL_ROLE: ${ secrets.SNOWFLAKE_ROLE }
23       SNOWFLAKE_WAREHOUSE: ${ secrets.SNOWFLAKE_WAREHOUSE }
24
25     steps:
26       - name: Checkout repository
27         uses: actions/checkout@v3
28
29       - name: Install SnowSQL
30         run: |
31           curl -O https://sfc-repo.snowflakecomputing.com/snowsql/bootstrap/1.2/linux_x86_64/snowsql-1.2.9-linux_x86_64.bash
32           SNOWSQL_DEST=~/.bin SNOWSQL_LOGIN_SHELL=~/.profile bash snowsql-1.2.9-linux_x86_64.bash
33
34       - name: Deploy Create Stages
35         run: |
36           ~/bin/snowsql -f dev_poc_sap/POC_SAP_LANDING/DEV_CREATE_STAGES.sql -o friendly=false
37
38       - name: Deploy Git Integration Object
39         run: |
40           ~/bin/snowsql -f dev_poc_sap/POC_SAP_LANDING/DEV_GIT_INTEGRATION.sql -o friendly=false
41
42       - name: Deploy File Formats JSON
43         run: |
44           ~/bin/snowsql -f dev_poc_sap/POC_SAP_LANDING/DEV_FILE_FORMAT_JSON_1.0.0.sql -o friendly=false
45
46       - name: Deploy File Formats CSV
47         run: |
48           ~/bin/snowsql -f dev_poc_sap/POC_SAP_LANDING/DEV_FILE_FORMATS_CSV_2.0.0.sql -o friendly=false
```

4.

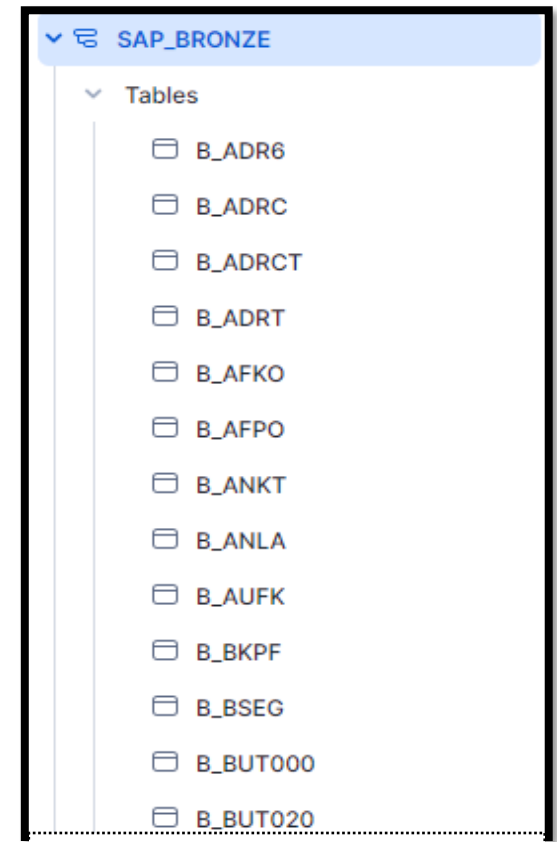
LANDING TO BRONZE

LANDING TO BRONZE

Structure du schéma SAP_BRONZE

MAIN INTEGRATIONS

- Création des Tables SAP Techniques ✓
- Change Data Capture (CDC) Columns -> **operation_flag** , **is_deleted** ✓
- Historical Data Tracking -> **Recordstamp** ✓
- Alimentation des Tables (>>100) assurée -> **COPY INTO – Snowpark**



Src : Snowflake
Database : DEV_DB_VISEO
Schema : SAP_BRONZE

LANDING TO BRONZE

Process de passage du Landing To Bronze

Query History

Status All User VISEO X Last 14 days Filters 1000+ Queries Columns

SQL TEXT	QUERY ID	STATUS	USER	WAREHOUSE	DURATION	STARTED
CALL DEV_DB_VISEO.SAP_RAW_LANDING.BULK_LOAD_SP();	01b51765-0000-4eec-0000-c7a90001dd2a	Success	VISEO	DEV_WH	11m 53s	6/18/2024, 2:21:18 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_VBFA FR...	01b5176d-0000-4eec-0000-c7a90002e2d2	Success	VISEO	DEV_WH	492ms	6/18/2024, 2:29:17 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_VBEP FR...	01b5176d-0000-4eec-0000-c7a90002e2c6	Success	VISEO	DEV_WH	501ms	6/18/2024, 2:29:16 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_VBAP FR...	01b5176d-0000-4eec-0000-c7a90002e2ba	Success	VISEO	DEV_WH	739ms	6/18/2024, 2:29:15 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_VBAK FR...	01b5176c-0000-4eec-0000-c7a90002e2ae	Success	VISEO	DEV_WH	38s	6/18/2024, 2:28:37 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVMT FR...	01b5176c-0000-4eec-0000-c7a90002e2a2	Success	VISEO	DEV_WH	475ms	6/18/2024, 2:28:36 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVMT FR...	01b5176c-0000-4eec-0000-c7a90002e296	Success	VISEO	DEV_WH	519ms	6/18/2024, 2:28:35 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVLST FR...	01b5176c-0000-4eec-0000-c7a90002e28a	Success	VISEO	DEV_WH	514ms	6/18/2024, 2:28:34 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVKT FR...	01b5176c-0000-4eec-0000-c7a90002e27e	Success	VISEO	DEV_WH	487ms	6/18/2024, 2:28:34 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVKO FR...	01b5176c-0000-4eec-0000-c7a90002e272	Success	VISEO	DEV_WH	491ms	6/18/2024, 2:28:33 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVFS FR...	01b5176c-0000-4eec-0000-c7a90002e266	Success	VISEO	DEV_WH	469ms	6/18/2024, 2:28:32 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVFS FR...	01b5176c-0000-4eec-0000-c7a90002e25a	Success	VISEO	DEV_WH	484ms	6/18/2024, 2:28:31 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TVAVC FR...	01b5176c-0000-4eec-0000-c7a90002e24e	Success	VISEO	DEV_WH	703ms	6/18/2024, 2:28:31 PM
COPY INTO DEV_DB_VISEO.SAP_BRONZE.B_TSPAT FR...	01b5176c-0000-4eec-0000-c7a90002e242	Success	VISEO	DEV_WH	502ms	6/18/2024, 2:28:30 PM

Src : Snowflake – Query History



Procedure definition

```
1 CREATE OR REPLACE PROCEDURE DEV_DB_VISEO.SAP_RAW_LANDING.BULK_LOAD_SP()
2 RETURNS VARCHAR(16777216)
3 LANGUAGE PYTHON
4 RUNTIME_VERSION = '3.8'
5 PACKAGES = ('snowflake-snowpark-python')
6 HANDLER = 'DEV_BULK_LOAD_CSV_TO_TABLES.load_data_automated'
7 IMPORTS = ('@DEV_DB_VISEO.SAP_RAW_LANDING.GIT_REPO_STAGE/branches/main/dev_poc_sap/POC_SAP_LANDING/DEV_BULK_LOAD_CSV_TO_TABLES.py')
8 EXECUTE AS OWNER
9 ;
```

Show less

Src : Snowflake – Procedure Definition

1

Procedure definition

```
1 CREATE OR REPLACE PROCEDURE DEV_DB_VISEO.SAP_RAW_LANDING.CREATE_BRONZE_SP()
2 RETURNS VARCHAR(16777216)
3 LANGUAGE PYTHON
4 RUNTIME_VERSION = '3.8'
5 PACKAGES = ('snowflake-snowpark-python')
6 HANDLER = 'DEV_CREATE_TABLES_BRONZE.create_tables'
7 IMPORTS = ('@DEV_DB_VISEO.SAP_RAW_LANDING.GIT_REPO_STAGE/branches/main/dev_poc_sap/POC_SAP_LANDING/DEV_CREATE_TABLES_BRONZE.py')
8 EXECUTE AS OWNER
9 ;
```

Show less

Src : Snowflake – Procedure Definition

poc-sap / dev_poc_sap / POC_SAP_LANDING / DEV_BULK_LOAD_CSV_TO_TABLES.py

eddaouissam Release 1.0 - Introducing Git Integration Feature

Code Blame 64 lines (55 loc) · 3.32 KB

```
1 """
2 Script: Automated Data Loader
3 Author: M'hamed Issam ED-DAOU
4 Description:
5 This Snowpark Python script automates the process of loading data from specified files in a Snowflake stage into corresponding tables.
6 It accounts for variations in naming conventions (e.g., underscores in table names but not in file names) by normalizing the names during pr
7 This functionality is crucial for environments with frequent data updates and where automated data loading needs to manage minor discrepanci
8 """
9
10 from snowflake.snowpark import Session
11
12 def load_data_automated(session: Session) -> str:
13     """
14     Automatically loads data from files in a specified stage into tables with corresponding names in Snowflake,
15     ignoring any 'B_' prefixes in the table names.
16
17     Parameters:
18     session (snowflake.snowpark.Session): The session object for database interaction.
19
20     Returns:
21     str: A message indicating the overall status of the load operations.
22     """
23     stage_name = 'DEV_DB_VISEO.SAP_RAW_LANDING.SAP_CSV_FILES'
24     file_format_name = 'DEV_DB_VISEO.SAP_RAW_LANDING.FF_CSV_SAP'
25     target_schema = 'DEV_DB_VISEO.SAP_BRONZE'
26     messages = []
```

Src : Git Repo – DEV BULK LOAD CSV TO TABLES.py

2

LANDING TO BRONZE

Process de passage du Landing To Bronze Orchestré



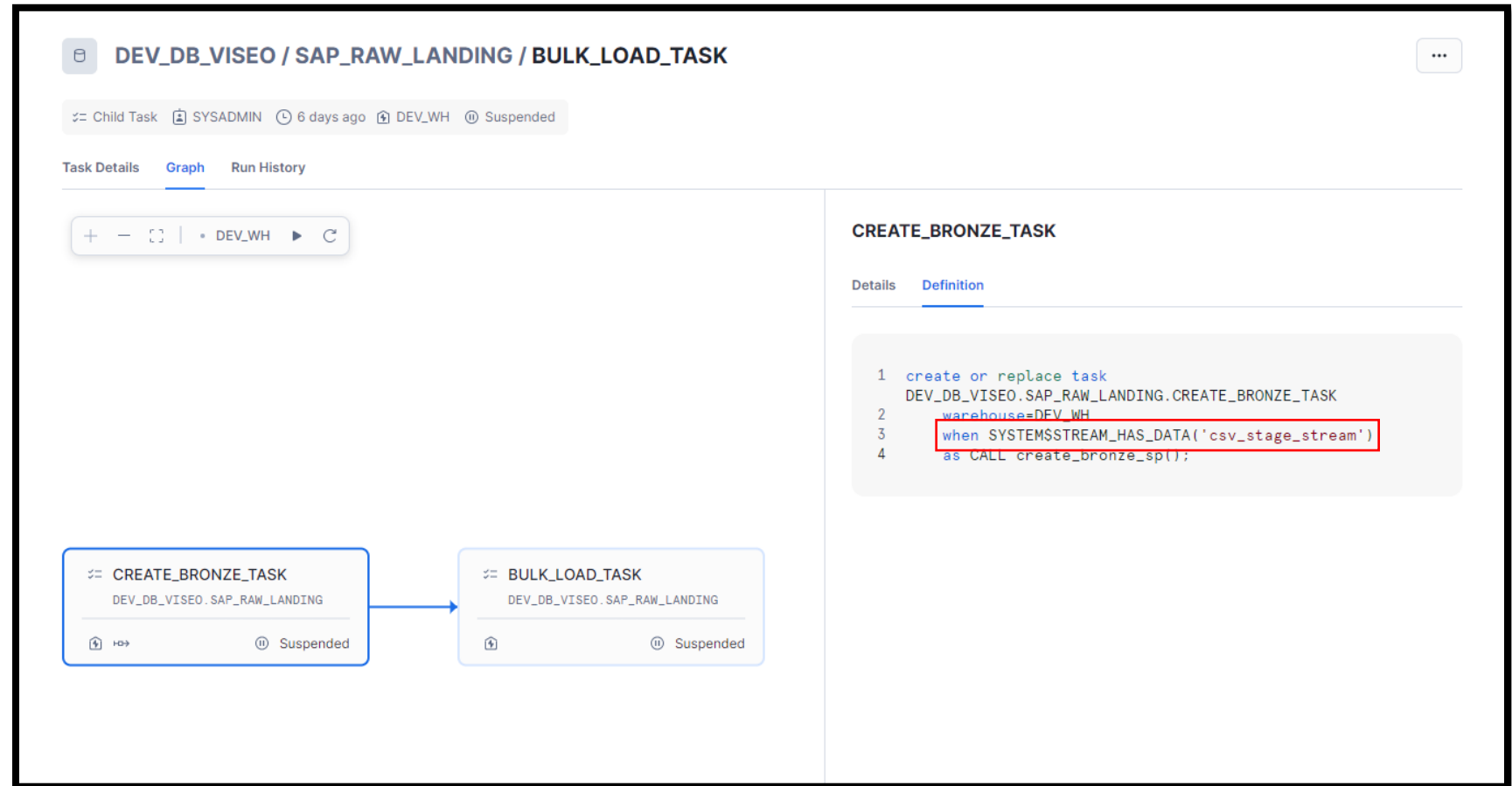
Snowflake Streams ?

Fonctionnalité permettant de suivre les modifications apportées aux tables, vues, External Tables, et **Directory Tables**



Snowflake Tasks ?

Les Tasks sont des objets qui exécutent une seule commande **SQL** ou un call une **procédure stockée** selon un **schedule/event**.



Src : Snowflake – Task Graph

5.

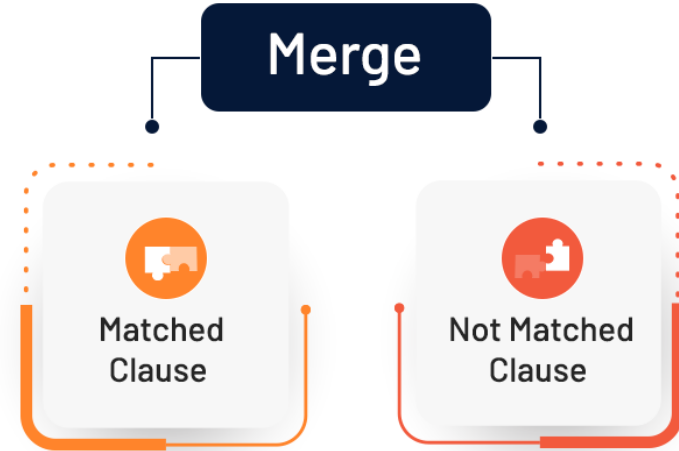
BRONZE TO SILVER

BRONZE TO SILVER

MERGE QUERY

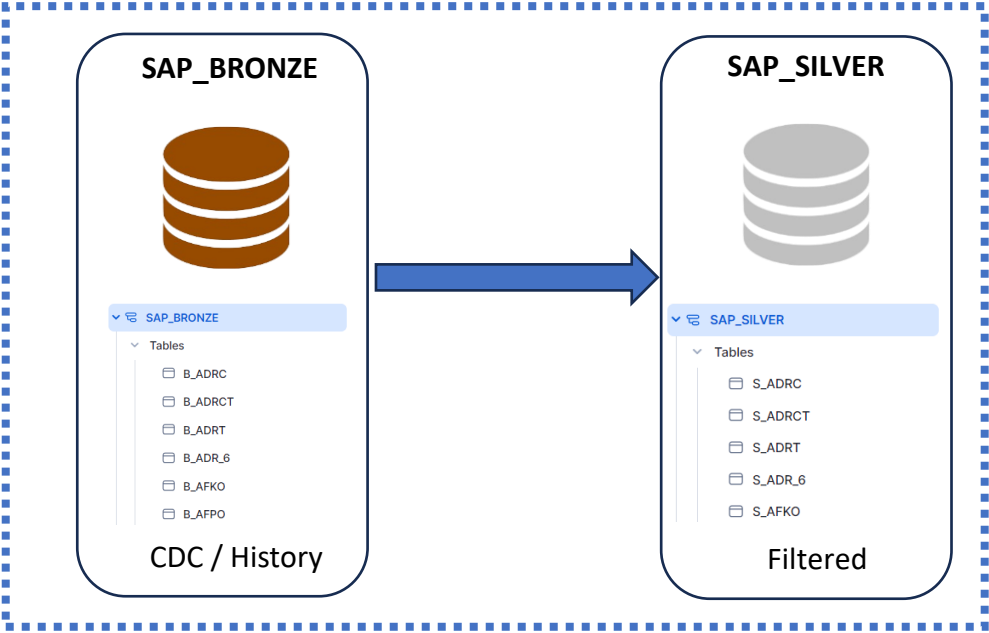


- CDC operations: Gestion des **inserts**, **updates**, and **deletes** basé sur --> **operation_flag**, **is_deleted** ✓
- Filtrer les données historiques (colonne du **recordstamp**) ✓
- gestion des Duplicates: --> **ROW_NUMBER()** ✓



BRONZE TO SILVER

Process de passage du Bronze To Silver



Query History

Status: All | User: VISEO | Last 14 days | Filters | 1000+ Queries

SQL TEXT	QUERY ID	STATUS	USER	WAREHOUSE	DURATION
CALL SAP_MERGE();	01b51772-0000-4f2e-0000-c7a90002b0f6	Success	VISEO	DEV_WH	7m 15s
MERGE INTO SAP_SILVER.S_T161 AS T USING (SE...	01b51779-0000-4f2e-0000-c7a90002bad2	Success	VISEO	DEV_WH	629ms
SELECT fieldname FROM SAP_BRONZE.B_dd03L.WHER...	01b51779-0000-4f2e-0000-c7a90002bace	Success	VISEO	DEV_WH	313ms
SELECT column_name FROM information_schema.c...	01b51779-0000-4f2e-0000-c7a90002baca	Success	VISEO	DEV_WH	958ms
MERGE INTO SAP_SILVER.S_VBRP AS T USING (SE...	01b51779-0000-4f2e-0000-c7a90002babe	Success	VISEO	DEV_WH	5.7s
SELECT fieldname FROM SAP_BRONZE.B_dd03L.WHER...	01b51779-0000-4f2e-0000-c7a90002baba	Success	VISEO	DEV_WH	296ms
SELECT column_name FROM information_schema.c...	01b51779-0000-4f2e-0000-c7a90002bab6	Success	VISEO	DEV_WH	978ms

Exemple de Transformation

B_ADR6	11 Rows
ADDRNUMBER	VARCHAR(16777216)
CLIENT	VARCHAR(16777216)
CONSNUMBER	VARCHAR(16777216)
DATE_FROM	DATE
DFT_RECEIV	VARCHAR(16777216)
ENCODE	VARCHAR(16777216)
FLG_NOUSE	VARCHAR(16777216)
FLGDEFAULT	VARCHAR(16777216)
ADDRNUMBER	VARCHAR(16777216)
HOME_FLAG	BOOLEAN
IS_DELETED	BOOLEAN
OPERATION_FLAG	BOOLEAN
PERSNUMBER	VARCHAR(16777216)
R3_USER	VARCHAR(16777216)
RECORDSTAMP	TIMESTAMP_NTZ(9)
SMTP_ADDR	VARCHAR(16777216)
SMTP_SRCH	VARCHAR(16777216)
TNEF	VARCHAR(16777216)
VALID_FROM	VARCHAR(16777216)
VALID_TO	VARCHAR(16777216)

S_ADR6	8 Rows
CONSNUMBER	VARCHAR(16777216)
RECORDSTAMP	TIMESTAMP_NTZ(9)
TNEF	VARCHAR(16777216)
DFT_RECEIV	VARCHAR(16777216)
HOME_FLAG	BOOLEAN
FLGDEFAULT	BOOLEAN
ADDRNUMBER	VARCHAR(16777216)
R3_USER	VARCHAR(16777216)
ENCODE	VARCHAR(16777216)
CLIENT	VARCHAR(16777216)
VALID_TO	VARCHAR(16777216)
SMTP_SRCH	VARCHAR(16777216)
FLG_NOUSE	VARCHAR(16777216)
DATE_FROM	DATE
SMTP_ADDR	VARCHAR(16777216)
VALID_FROM	VARCHAR(16777216)
PERSNUMBER	VARCHAR(16777216)

BRONZE TO SILVER

MERGE Query : Data Quality Check

The screenshot shows the Snowflake Merge Query Checker interface. The left sidebar displays a tree view of databases and worksheets. The main area shows a Python script for a CDC consistency checker. The script is titled 'Script: Automated CDC Consistency Checker' and is authored by 'M'hamed Issam ED-DAOU'. It describes a Snowpark Python script designed to validate the CDC process between two schemas in Snowflake: SAP_BRONZE (source) and SAP_SILVER (target). The script verifies the integrity and consistency of data transferred from the source to the target schema by comparing the row counts of tables after applying CDC operations. It also generates logs for each table it checks, offering insights into potential discrepancies and confirming successful synchronization.

```
1  """
2  Script: Automated CDC Consistency Checker
3  Author: M'hamed Issam ED-DAOU
4  Description:
5  This Snowpark Python script is designed to validate the Change Data Capture (CDC) process between
6  two schemas in Snowflake: SAP_BRONZE (source) and SAP_SILVER (target). The script verifies the integrity
7  and consistency of data transferred from the source to the target schema by comparing the row counts
8  of tables after applying CDC operations.
9
10 This tool generates logs for each table it checks, offering insights into potential discrepancies
11 and confirming successful synchronization, thereby aiding in troubleshooting and auditing CDC processes.
12 """
13
14 from snowflake.snowpark.session import Session
15
16 def get_primary_keys(session: Session, base_table_name: str) -> list:
17     """ Fetch primary keys from B_dd03L for a given table """
18     primary_keys_query = f"""
19         SELECT fieldname
20         FROM SAP_BRONZE.B_dd03L
21         WHERE UPPER(tablename) = UPPER('{base_table_name}') AND keyflag = 'X' AND fieldname != '.INCLUDE'
22     """
23     return session.sql(primary_keys_query).collect()
```

The bottom section shows the results of the query, indicating 'ALL SUCCESS !!!!'.

Src : Snowflake – Worksheets

The screenshot shows the Snowflake Results tab, displaying the output of the data quality check. The output is a list of checks performed on various tables, comparing the latest records from the source (SAP_BRONZE) against the target (SAP_SILVER). The checks include:

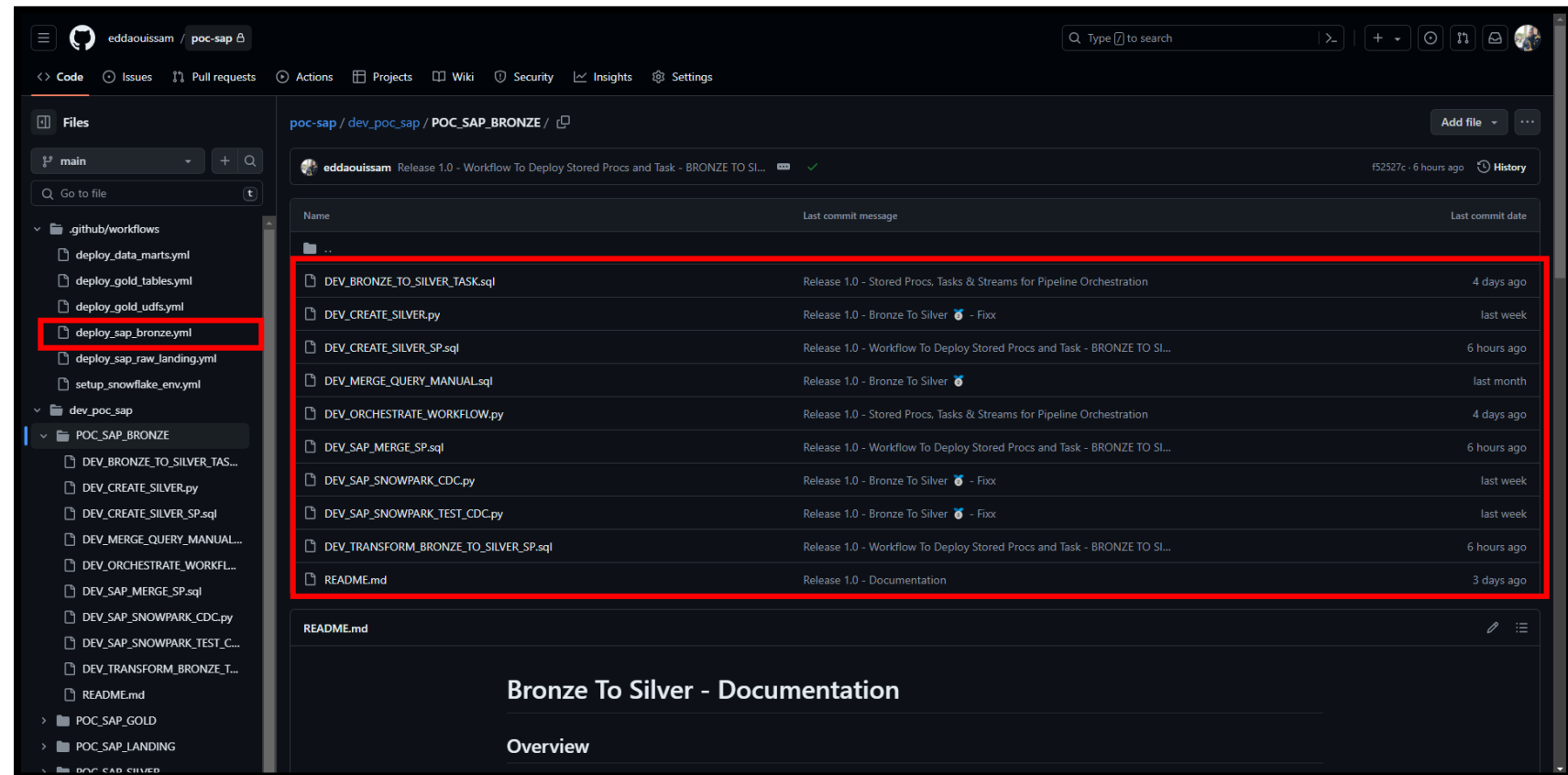
- Checking B_ADR6 against SAP_SILVER.S_ADR6: Rows in latest records from B_ADR6: 8, Rows in SAP_SILVER.S_ADR6: 8
- Checking B_ADRCT against SAP_SILVER.S_ADRCT: Rows in latest records from B_ADRCT: 13231, Rows in SAP_SILVER.S_ADRCT: 13231
- Checking B_ADRT against SAP_SILVER.S_ADRT: Rows in latest records from B_ADRT: 31626, Rows in SAP_SILVER.S_ADRT: 31626
- Checking B_AFPO against SAP_SILVER.S_AFPO: Rows in latest records from B_AFPO: 0, Rows in SAP_SILVER.S_AFPO: 0
- Checking B_ANKT against SAP_SILVER.S_ANKT: Rows in latest records from B_ANKT: 1607, Rows in SAP_SILVER.S_ANKT: 1607
- Checking B_ANLA against SAP_SILVER.S_ANLA: Rows in latest records from B_ANLA: 1487, Rows in SAP_SILVER.S_ANLA: 1487
- Checking B_AUFK against SAP_SILVER.S_AUFK: Rows in latest records from B_AUFK: 4, Rows in SAP_SILVER.S_AUFK: 4
- Checking B_BKPF against SAP_SILVER.S_BKPF: Rows in latest records from B_BKPF: 1364810, Rows in SAP_SILVER.S_BKPF: 1364810
- Checking B_BSEG against SAP_SILVER.S_BSEG: Rows in latest records from B_BSEG: 2105358, Rows in SAP_SILVER.S_BSEG: 2105358

Src : Snowflake –
Worksheets/Logs

BRONZE TO SILVER

Démo Git Repo

- Orchestration du process ✓
- Automatisation du déploiement ✓
- Documentation ✓



Git Repo : poc-sap/dev_poc_sap/POC_SAP_BRONZE

6.

SILVER TO GOLD

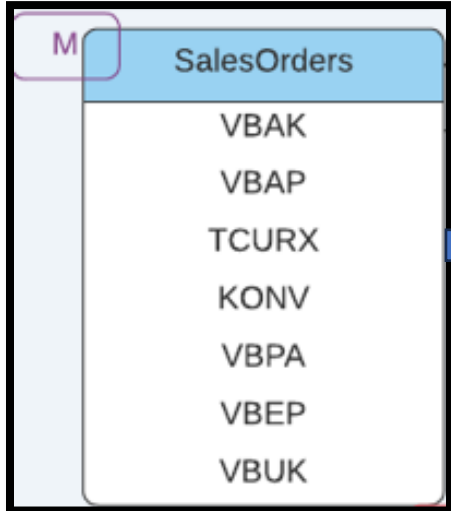
SILVER TO GOLD

Option 1

Exemple de process d'Adaptation des scripts des Tables Fonctionnelles



Visual Studio Code

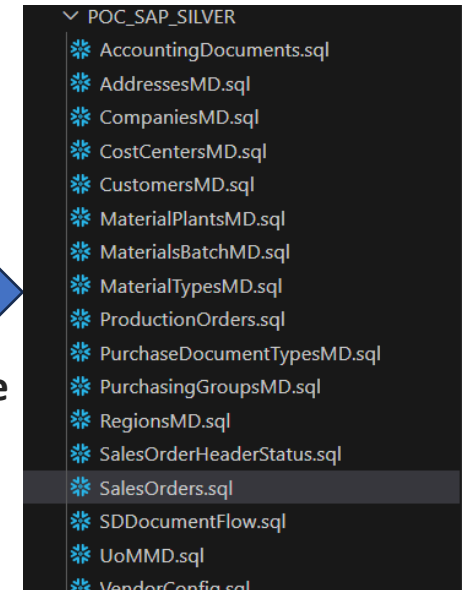


Modèle E-R

```
1 WITH TCURX AS (  
2   -- Joining to this table is necessary to fix the decimal place of  
3   -- amounts for non-decimal-based currencies. SAP stores these amounts  
4   -- offset by a factor of 1/100 within the system (FYI this gets  
5   -- corrected when a user observes these in the GUI) Currencies w/  
6   -- decimals are unimpacted.  
7   --  
8   -- Example of impacted currencies JPY, IDR, KRW, THD  
9   -- Example of non-impacted currencies USD, GBP, EUR  
10  -- Example 1,000 JPY will appear AS 10.00 JPY  
11  SELECT DISTINCT  
12    CURRKEY,  
13    CAST(POWER(10, 2 - COALESCE(CURRDEC, 0)) AS NUMERIC) AS CURRFX  
14  FROM  
15    '{{ project_id_src }}.{{ dataset_cdc_processed_ecc }}.tcuxr'  
16  ),  
17  
18  AGGKONV AS (  
19    SELECT  
20      KONV.KNRMW,  
21      KONV.KPOSN,  
22      KONV.MANDT,
```

Src : Google Cortex Framework

Python +
Validation manuelle



Repo de notre Projet
SNOWFLAKE

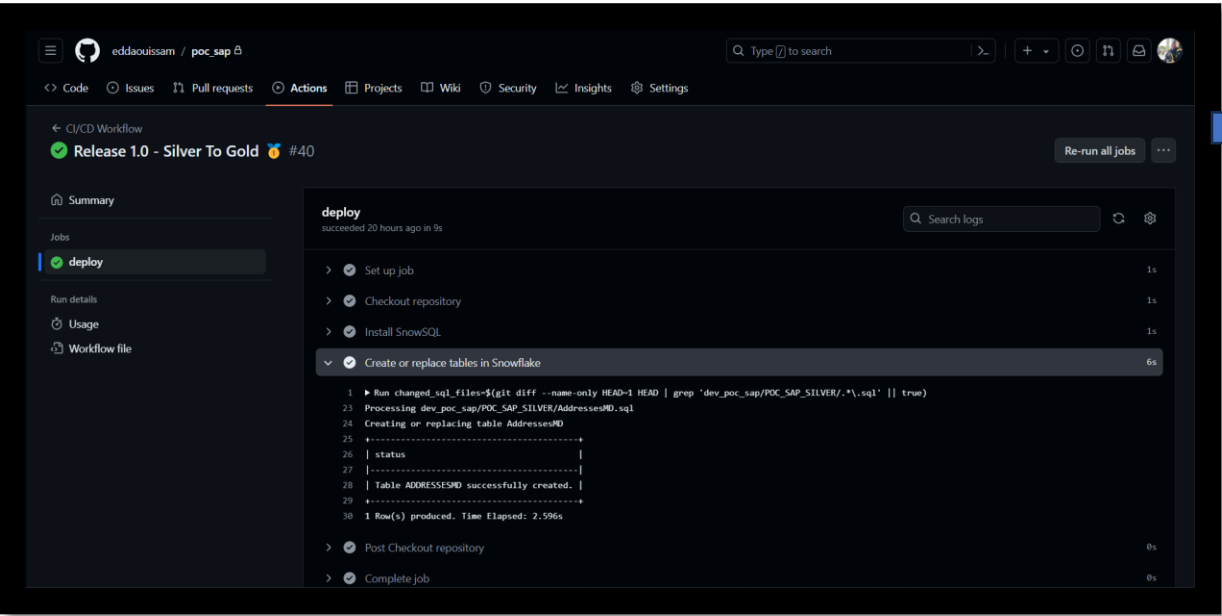
SILVER TO GOLD

Option 1

Process du Déploiement vers Snowflake – DML TO DDL



Workflow Files :
deploy_gold_udfs.yml ✓
deploy_gold_tables.yml ✓



Git Repo : CI/CD Workflow

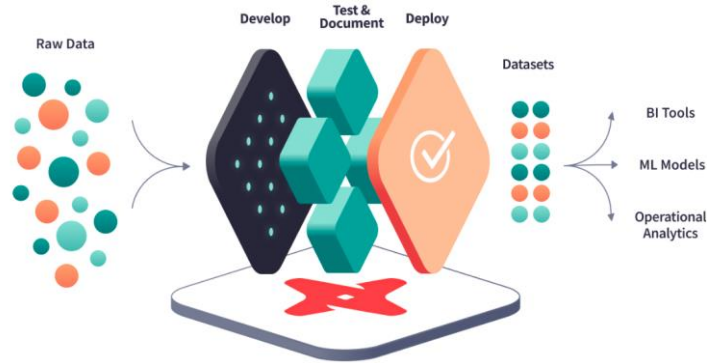


Query History							
Status All		User VISEO	Last 14 days	Filters	1000+ Queries		
SQL TEXT	D	STATUS	USER	WAREHOUSE	DURATI...	STARTED	CLIENT DRIVER
CREATE OR REPLACE TABLE SAP_GOLD.Pr...	7-0000-4a4d-0000-b9650...	Succe...	VIS...	DEV_WH	1.9s	5/23/2024, 3:43:53 PM	SnowSQL 1.2.32
CREATE OR REPLACE TABLE SAP_GOLD.Cu...	7-0000-4a6e-0000-b9650...	Succe...	VIS...	DEV_WH	1.3s	5/23/2024, 3:43:50 PM	SnowSQL 1.2.32
CREATE OR REPLACE TABLE SAP_GOLD.Co...	7-0000-4a6e-0000-b9650...	Succe...	VIS...	DEV_WH	1.4s	5/23/2024, 3:43:47 PM	SnowSQL 1.2.32

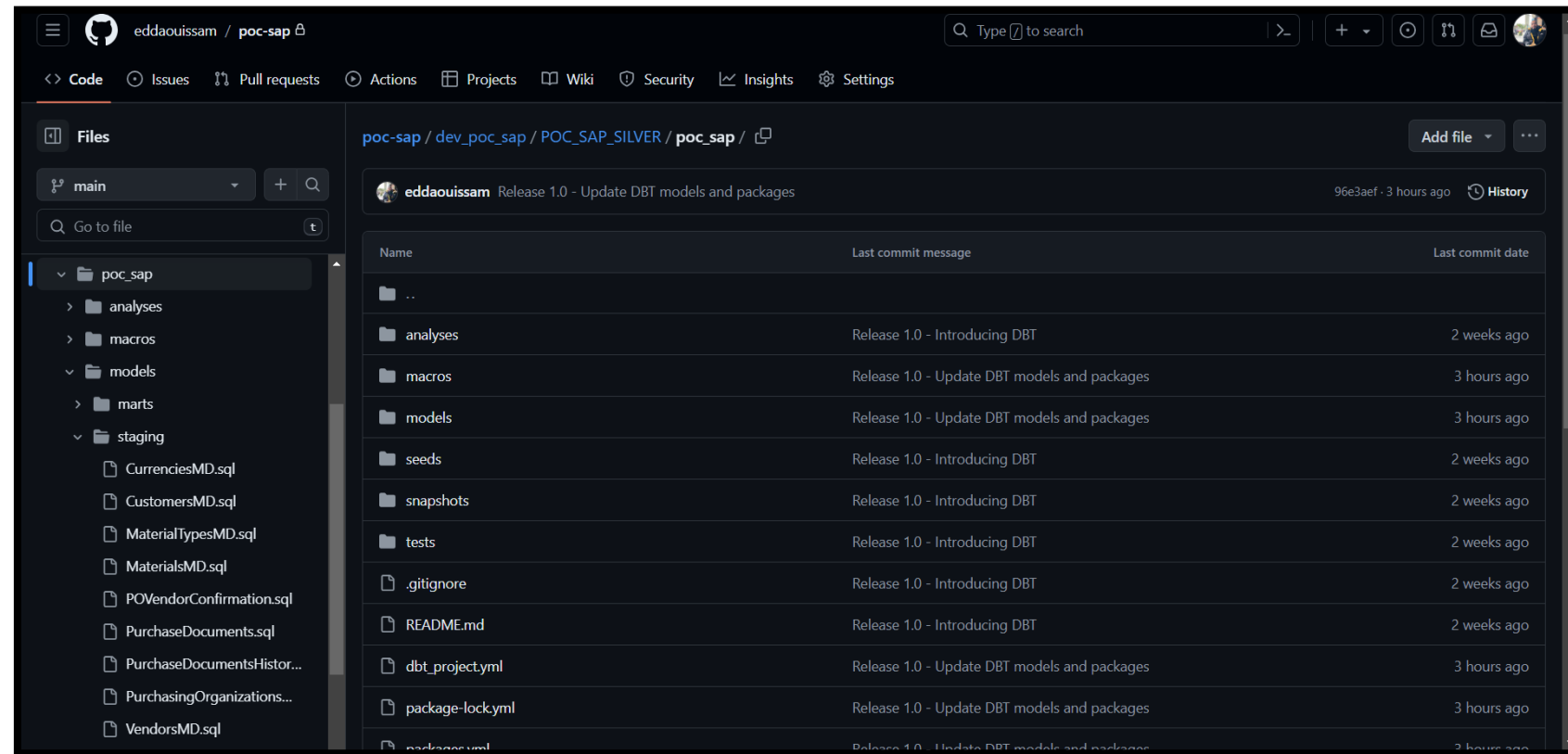
Src : Snowflake – Query History

SILVER TO GOLD

Intégration du DBT (Data Build Tool)



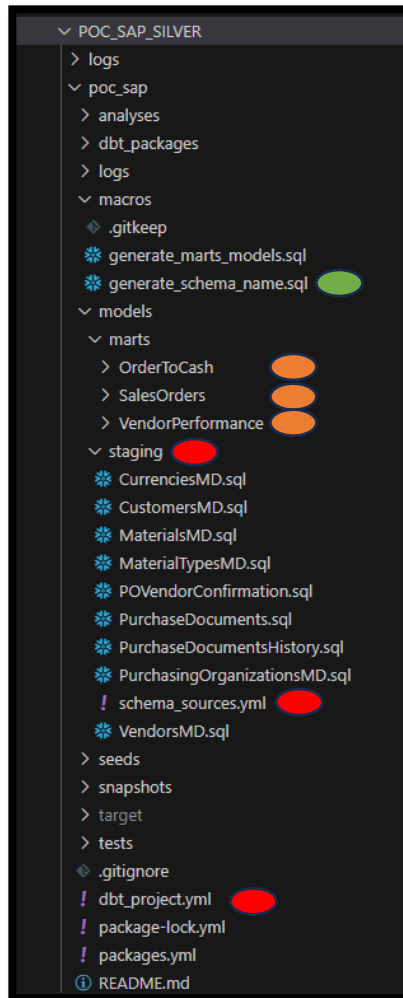
- Tests ✓
- Documentation ✓
- Data Lineage ✓
- Interopérabilité avec ❄️ ✓



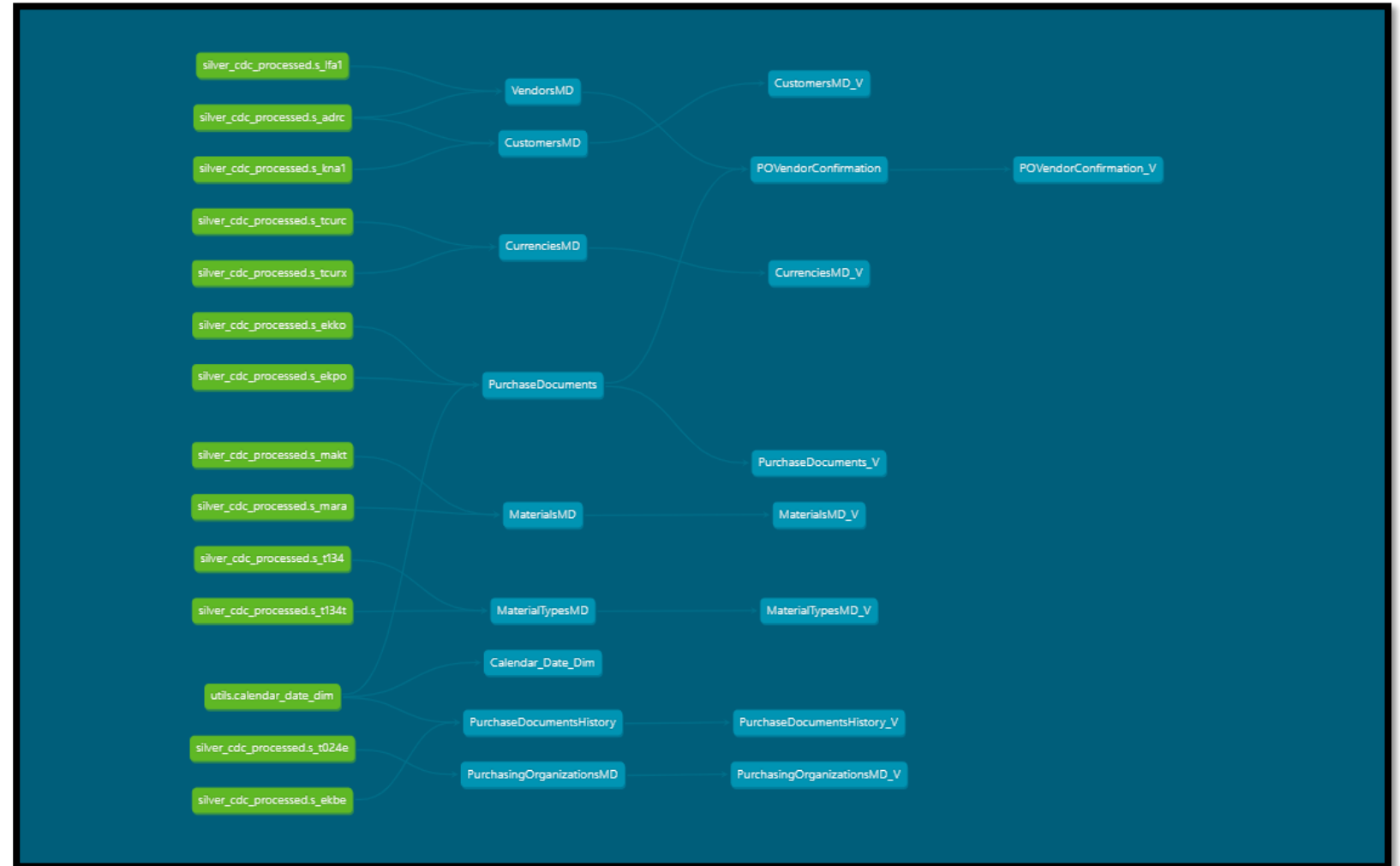
Git Repo : poc-sap/dev_poc_sap/POC_SAP_SILVER/poc_sap

SILVER TO GOLD

DBT (Data Build Tool)



Structure du projet DBT ~



Data Lineage – DBT Documentation – En cours~~



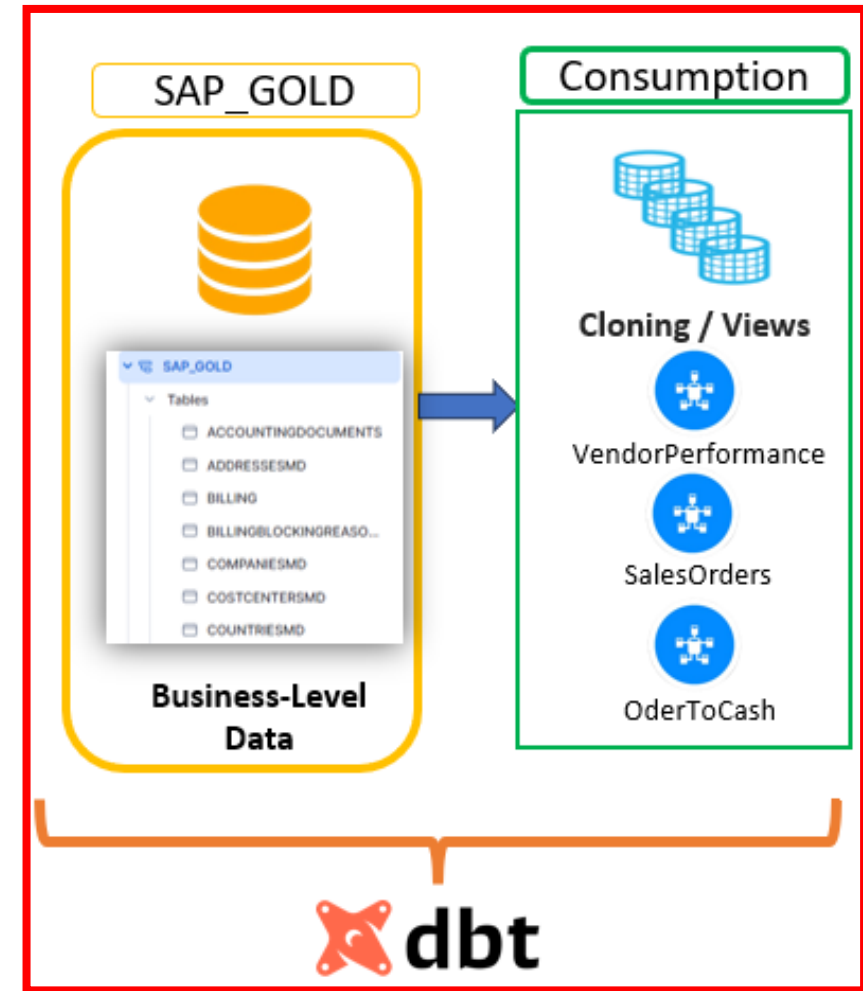
7.

Conclusion et Prochaines étapes

CONCLUSION

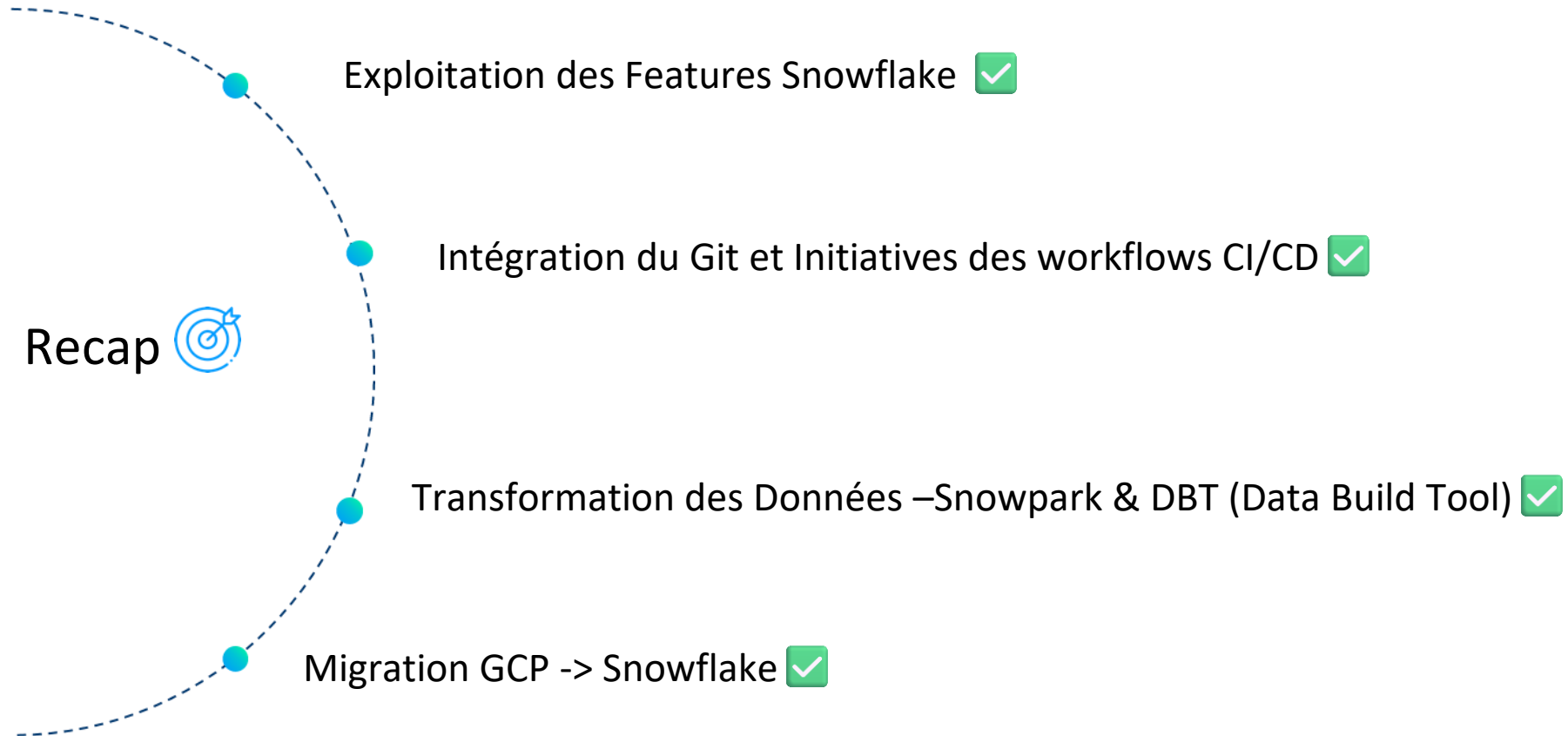
Étapes actuelles/prochaines

- Continuer à adapter les scripts Cortex pour Snowflake **SAP_GOLD**
- Data Modeling : E-R-D -> Star Schemas (pour les **Data Marts**)
- **CI/CD** pour la phase du DBT et **Orchestration**
- Git repo et DBT : **Documentation++** | **Structuration++**



CONCLUSION

Recap





UISEO

MERCI !