

Conception, Développement, et Exploitation d'un Data Lake pour une Entreprise Digitale

Bilgenur OZDEMIR
Xuan Thu NGUYEN
Boris Yves NZEUYO DJOKO
Jiayin CHEN

Partie 1 : Conception du Data Lake

1. Analyse des besoins et conception logique

a. Description :

L'objectif est de comprendre les besoins en données de l'entreprise et de concevoir un Data Lake adapté. Cela inclut :

- Identifier les sources de données (bases de données, fichiers, logs, API).
- Définir les types de données : structurées (SQL), semi-structurées (JSON, XML) et non structurées (images, vidéos, logs).
- Planifier l'organisation des données en zones :
 - **Zone brute** (Zone brute) : Données telles qu'ingérées. : gs://data-lake-project-raw/
Les dossiers suivants ont été créés pour organiser les données brutes par catégorie :
- Transactions : gs://data-lake-raw/transactions/Retail_Transaction_Dataset.csv
- Logs : gs://data-lake-raw/logs/apache_log.log
- Réseaux sociaux : gs://data-lake-raw/social/twitter.json
- Temps réel : gs://data-lake-raw/real-time/
- **Zone nettoyée** (Zone nettoyée) : Données après transformation. gs://data-lake-cleansed/
- **Zone analytique** (Zone analytique) : Données prêtes pour l'analyse. gs://data-lake-curated/

data-lake-project-raw

logs/

real-time/

scripts/

social/

transactions/

CRÉER UN DOSSIER

IMPORTER

TRANSFÉRER LES DONNÉES

AUTRES SERVICES

Filtrer par préfixe de nom uniquement

Filtrer les objets et dossiers

Afficher Objets actifs uniquement

<input type="checkbox"/>	Nom	Taille	Type	Création	Classe de stockage	Dernière modification	Accès public	Historique des versions	
<input type="checkbox"/>	Retail_Transaction_Dataset.csv	12,2 Mo	text/csv	25 nov. 2024, 14:14:05	Standard	25 nov. 2024, 14:14:05	Non public	—	

data-lake-project-raw

Zone

Classe de stockage

Accès public

Protection

us-central1 (Iowa)

Standard

Soumise à des LCA au niveau de l'objet

Supprimer de façon réversible

OBJET

CONFIGURATION

AUTORISATIONS

PROTECTION

CYCLE DE VIE

OBSERVABILITÉ

RAPPORTS SUR L'INVENTAIRE

OPÉRATIONS

Navigateur de dossiers

Buckets > data-lake-project-raw > social

CRÉER UN DOSSIER

IMPORTER

TRANSFÉRER LES DONNÉES

AUTRES SERVICES

Filtrer par préfixe de nom uniquement

Filtrer les objets et dossiers

Afficher Objets actifs uniquement

<input type="checkbox"/>	Nom	Taille	Type	Création	Classe de stockage	Dernière modification	Accès public	Historique des vers	
<input type="checkbox"/>	twitter.json	755 octets	application/json	25 nov. 2024, 14:40:01	Standard	25 nov. 2024, 14:40:01	Non public	—	

</

b. Technologies choisies sur GCP :

Nous avons choisi Google Cloud pour la réalisation de ce data lake. Cette infrastructure permettra une ingestion, un stockage, et un traitement efficace et scalable des données.

- Stockage avec GCS :
 - Création de buckets pour organiser les zones (Raw, Cleansed, Curated).
- Ingestion avec Pub/Sub :
 - Mise en place de topics et abonnements pour collecter les données en streaming.
- Traitement avec Dataproc :
 - Création d'un cluster Spark pour le traitement distribué.
- Analyse avec BigQuery :
 - Création de datasets pour l'analyse SQL des données nettoyées et agrégées.
- Sécurité : IAM (Identity and Access Management) pour contrôler les accès.

c. Diagramme des flux de données :

1. Ingestion :

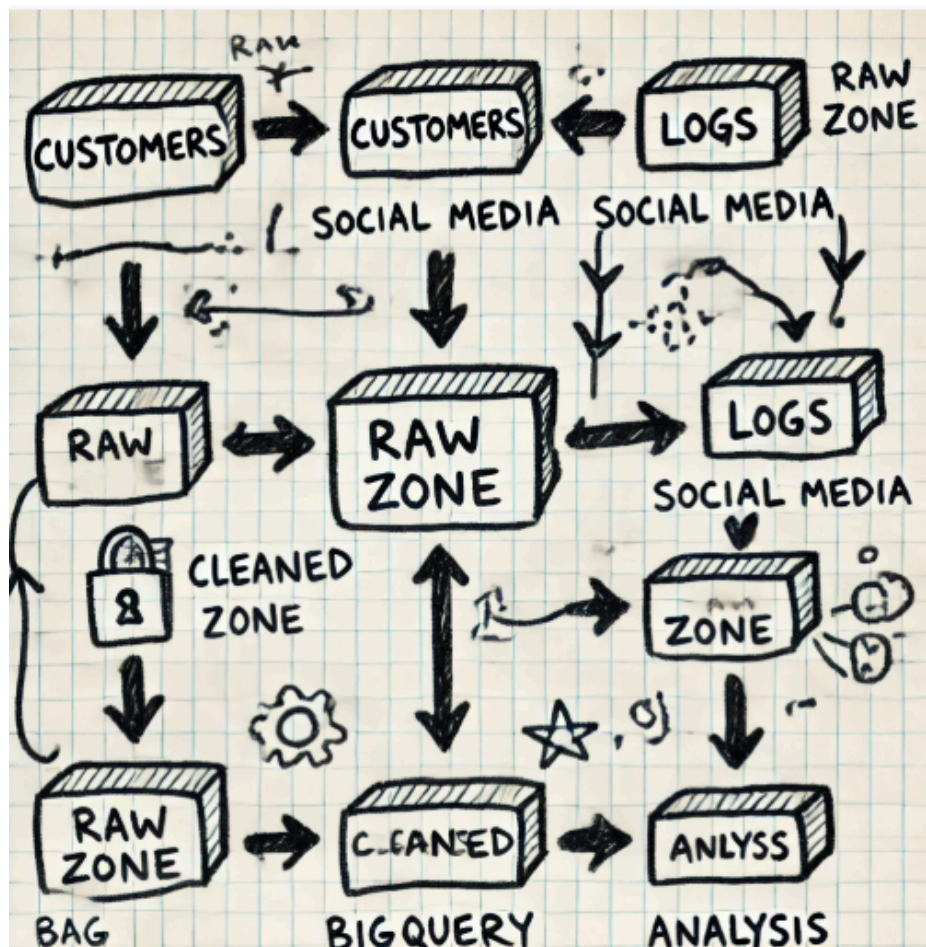
- Transactions clients (base SQL) → Cloud Storage (Raw Zone).
- Logs serveur (non structurés) → Cloud Storage (Raw Zone).
- Médias sociaux (API, JSON) → Cloud Storage (Raw Zone).
- Flux publicitaires → Pub/Sub → Raw Zone.

2. Transformation :

- Dataflow ou Spark pour nettoyer et enrichir → Cleansed Zone.

3. Analyse :

- Chargement des données cleansed dans BigQuery pour des analyses.



2.Création de l'infrastructure

Pour la création de l'infrastructure, nous avons utilisé le terminal (Cloud Shell) sur GCP, ci-dessous le fichier shell bash :

```
#!/bin/bash

# Variables
PROJECT_ID="data-lake-project"
```

```
LOCATION="us-central1"

# Étape 1 : Créer un projet GCP et configurer
echo "Création du projet GCP..."
gcloud projects create $PROJECT_ID --set-as-default
gcloud config set project $PROJECT_ID

# Étape 2 : Activer les API nécessaires
echo "Activation des API nécessaires..."
gcloud services enable storage.googleapis.com pubsub.googleapis.com
dataproc.googleapis.com bigquery.googleapis.com

# Étape 3 : Créer les buckets dans Google Cloud Storage
echo "Création des buckets..."
gsutil mb -p $PROJECT_ID -l $LOCATION gs://data-lake-raw/
gsutil mb -p $PROJECT_ID -l $LOCATION gs://data-lake-cleansed/
gsutil mb -p $PROJECT_ID -l $LOCATION gs://data-lake-curated/

# Étape 4 : Organiser les dossiers dans le bucket RAW
echo "Organisation des dossiers dans la zone RAW..."
gsutil mkdir gs://data-lake-raw/transactions/
gsutil mkdir gs://data-lake-raw/logs/
gsutil mkdir gs://data-lake-raw/social/
gsutil mkdir gs://data-lake-raw/real-time/

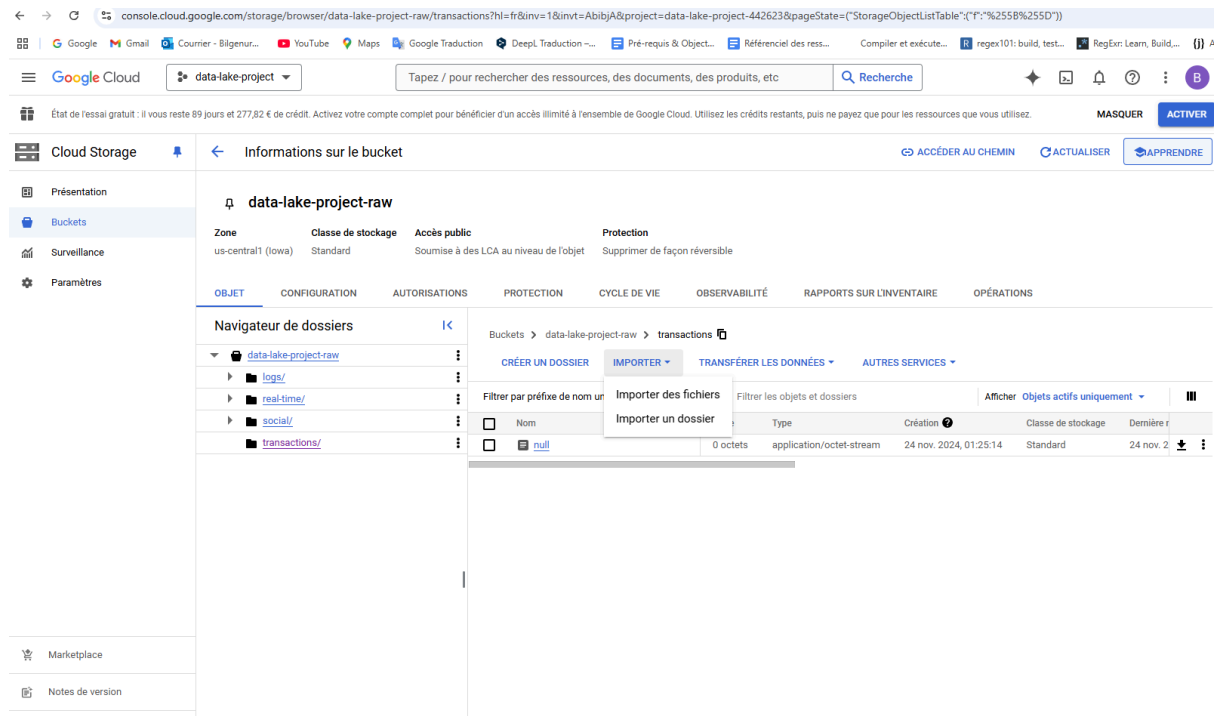
# Étape 5 : Vérifier la création des buckets et dossiers
echo "Vérification des buckets et dossiers créés..."
gsutil ls -p $PROJECT_ID

# Étape 6 : Importer les données dans les buckets
echo "Importation des données..."

#import d'un fichier dans le dossier transactions
gsutil cp "D:/bilge/M2/Data Lakes Lionel/Nouveau dossier
(2)/Retail_Transaction_Dataset.csv"
gs://data-lake-project-raw/transactions/
gsutil cp "D:/bilge/M2/Data Lakes Lionel/Nouveau dossier
(2)/apache_logs.log" gs://data-lake-project-raw/log/
gsutil cp "D:/bilge/M2/Data Lakes Lionel/Nouveau dossier
(2)/twitter.json" gs://data-lake-project-raw/social/

echo "Infrastructure Data Lake déployée avec succès !"
```

3.Import de fichier CSV



```
bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623) $ gsutil ls -r gs://data-lake-project-raw/
gs://data-lake-project-raw/logs/:
gs://data-lake-project-raw/logs/apache_logs.log

gs://data-lake-project-raw/real-time/:
gs://data-lake-project-raw/real-time/null

gs://data-lake-project-raw/social/:
gs://data-lake-project-raw/social/twitter.json

gs://data-lake-project-raw/transactions/:
gs://data-lake-project-raw/transactions/Retail_Transaction_Dataset.csv
```

Partie 2 : Ingestion et Transformation des Données

Étape 1 : Récupération de datasets publics

Voici des datasets que vous pouvez utiliser :

1. Transactions clients :
 - Récupéré sur Kaggle Dataset - Retail Data : gs://data-lake-raw/transactions/.

2. Logs serveur :
 - Récupéré sur github un fichier log
3. Données JSON (médias sociaux) :
 - Ecrit en exemple
4. Flux en temps réel :
 - Simulez des flux publicitaires via Pub/Sub :

```
bash
gcloud pubsub topics create flux_publicitaire
```

Fichier simulate_sub:

```
#Simuler des flux en temps réel avec Pub/Sub : Crée un simulateur de
messages Pub/Sub
from google.cloud import pubsub_v1

# Utiliser le bon identifiant de projet
project_id = "data-lake-project-442623"
topic_id = "flux_publicitaire"

publisher = pubsub_v1.PublisherClient()
topic_path = publisher.topic_path(project_id, topic_id)

# Publier des messages simulés
for i in range(10):
    data = f"AdEvent {i}".encode("utf-8")
    future = publisher.publish(topic_path, data)
    print(f"Published message ID: {future.result()}")
```

Résultat :

```
bilgenur_ozdemir95@cloudshell:~ (data-lake-project)$ gcloud config set project data-lake-project-442623
Updated property [core/project].
bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623)$ python simulate_pubsub.py
Published message ID: 12750909481811911
Published message ID: 12749021891053057
Published message ID: 12968533387887759
Published message ID: 12970419967293501
Published message ID: 12970406056148288
Published message ID: 12749468326203187
Published message ID: 12971041381382547
Published message ID: 12749697206216917
Published message ID: 12971194264781203
Published message ID: 12969815320146839
bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623)$
```

Pipelines de Transformation des Données avec cluster Dataproc

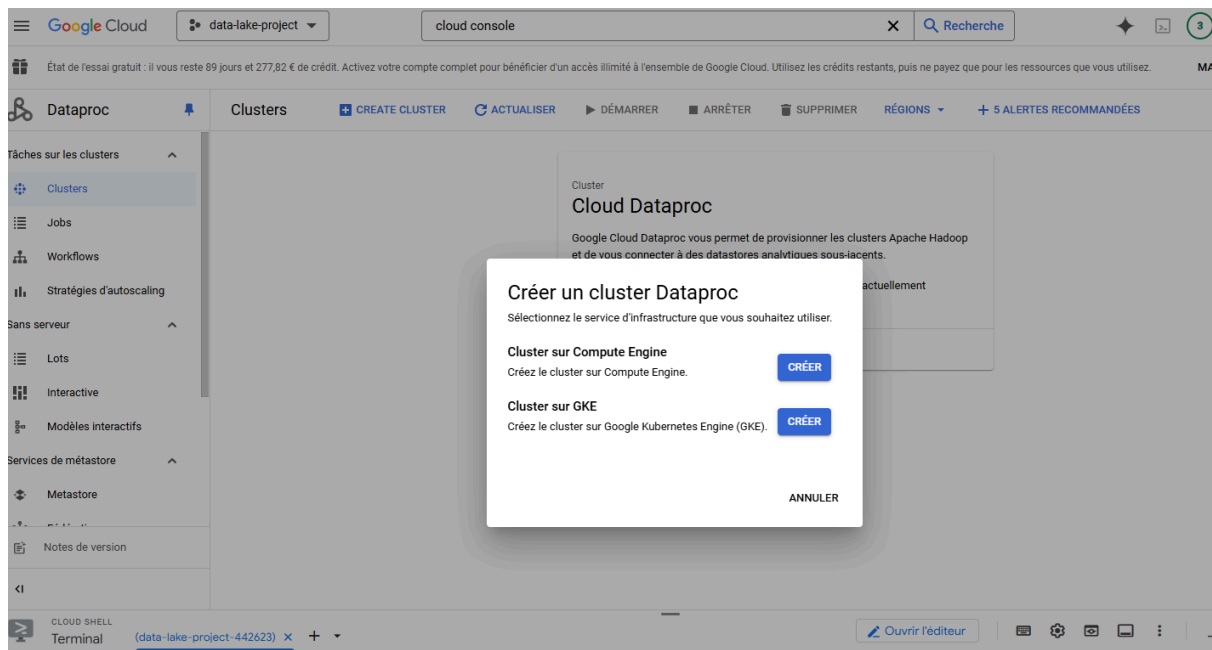
Dataproc est le service GCP pour gérer Spark et Hadoop.

Crée un cluster Spark/Hadoop avec Dataproc :

bash

Copier le code

```
gcloud dataproc clusters create data-lake-cluster \  
    --region us-central1 \  
    --zone us-central1-a \  
    --master-machine-type n1-standard-2 \  
--worker-machine-type n1-standard-2 \  
    --num-workers 2
```



Créer le cluster :

```
gcloud dataproc clusters create data-lake-cluster --enable-component-gateway --region  
us-central1 --zone us-central1-a --no-address --master-machine-type n1-standard-2  
--master-boot-disk-type pd-balanced --master-boot-disk-size 50 --num-workers 2  
--worker-machine-type n1-standard-2 --worker-boot-disk-type pd-balanced  
--worker-boot-disk-size 50 --project data-lake-project-442623
```

Maintenant que le cluster est en cours d'exécution créer un script PYSPARK pour nettoyer les données, transformer et créer un fichier parquet afin de les analyser :

```
from pyspark.sql import SparkSession  
import re
```

```
# Créer une session Spark
```

```

spark = SparkSession.builder.appName("DataCleaning").getOrCreate()

# Charger les données brutes
raw_data =
spark.read.text("gs://data-lake-project-raw/transactions/Retail_Transac
tion_Dataset.csv")

# Transformer les données en RDD pour un traitement ligne par ligne
rdd = raw_data.rdd.map(lambda row: row[0])

# Fonction pour détecter si une ligne commence par un chiffre
def starts_with_digit(line):
    return re.match(r"^\d", line)

# Fonction pour corriger les lignes mal alignées
def correct_misaligned_lines(lines):
    corrected = []
    buffer = None
    for line in lines:
        if starts_with_digit(line):
            if buffer:
                corrected.append(buffer)
            buffer = line
        else:
            if buffer:
                buffer += " " + line
    if buffer:
        corrected.append(buffer)
    return corrected

# Appliquer la correction sur les lignes
corrected_rdd = rdd.mapPartitions(correct_misaligned_lines)

# Convertir les lignes corrigées en DataFrame
schema = ["CustomerID", "ProductID", "Quantity", "Price",
"TransactionDate", "PaymentMethod", "StoreLocation", "ProductCategory",
"DiscountApplied", "TotalAmount"]

# Diviser chaque ligne corrigée en colonnes
data_split = corrected_rdd.map(lambda line: line.split(","))

# Vérifier que chaque ligne a exactement le nombre de colonnes attendu
data_filtered = data_split.filter(lambda row: len(row) == len(schema))

```



```
# Convertir les données filtrées en DataFrame
cleaned_df = data_filtered.toDF(schema=schema)

# Sauvegarder les données nettoyées au format Parquet
cleaned_df.write.parquet("gs://data-lake-cleansed/transactions_cleaned_v2.parquet", mode="overwrite")

# Afficher un aperçu des données nettoyées
cleaned_df.show(10)
```

Partie 2 : Logs

```
print("Nettoyage des logs...")
# Lire les logs Apache
logs =
spark.read.text("gs://data-lake-project-raw/logs/apache_logs.log")

# Transformer les logs pour extraire des colonnes utiles (parsing des lignes)
logs_transformed = logs.select(
    split(col("value"), " ")[0].alias("IP"),
    split(col("value"), " ")[3].alias("Timestamp"),
    split(col("value"), " ")[5].alias("Request_Type"),
    split(col("value"), " ")[6].alias("Endpoint")
)

# Sauvegarder les logs transformés
logs_transformed.write.parquet(
    "gs://data-lake-cleansed/logs_cleaned.parquet",
    mode="overwrite"
)

print("Nettoyage des logs terminé !")
```

Copier le fichier codé dans le scripts:

```
gsutil cp datacleaningv2.py gs://data-lake-project-raw/scripts/datacleaningv2.py
```

Executer ce spark :

```
bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623) $ gcloud dataproc jobs submit pyspark gs://data-lake-project-raw/scripts/datacleaningv2.py --cluster=my-dataproc-cluster --region=us-central1 --project=data-lake-project-442623
```

resultat avec aperçu 10 premieres valeur :

```

24/11/29 10:27:56 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at my-dataproc-cluster-m/10.128.0.5:8030
24/11/29 10:27:58 INFO com.google.cloud.hadoop.fs.gcs.GhfsStorageStatistics: Detected potential high latency for operation op_get_file_status. latencyMs=589; previousMaxLatencyMs=0; operationCount=1; context=gs://dataproc-temp-us-central1-1040140614716-leumlhg/615fb797-dac6-484e-9084-75f24e71d7df/spark-job-history
24/11/29 10:27:58 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object already exists with desired state.
24/11/29 10:28:49 INFO com.google.cloud.hadoop.repackaged.gcs.com.google.cloud.hadoop.gcsio.GoogleCloudStorageFilesystem: Successfully repaired 'gs://data-lake-cleansed/transactions_cleaned_v2.parquet/' directory.
+-----+-----+-----+-----+-----+-----+-----+-----+
|CustomerID|ProductID|Quantity|Price|TransactionDate|PaymentMethod|StoreLocation|ProductCategory|DiscountApplied|TotalAmount|
+-----+-----+-----+-----+-----+-----+-----+-----+
| 888163|D|7|13.12193739|12/26/2023 5:32|PayPal|USNV Harrell FPO...|Clothing|16.29512671|76.88590745|
| 609241|A|7|62.02823758|6/12/2023 7:02|PayPal|Unit 7268 Box 36...|Clothing|18.38875557|354.3541161|
| 147124|C|2|79.67386349|5/25/2023 10:49|Cash|USNS David FPO A...|Clothing|12.31370242|138.7699082|
| 210377|A|7|87.26431386|3/2/2024 22:56|Debit Card|Unit 4486 Box 34...|Books|5.36747935|578.0623889|
| 472995|B|5|69.2038385|5/19/2023 2:21|Cash|Unit 5493 Box 49...|Books|16.49585071|288.9403831|
| 509745|C|7|32.44361216|3/1/2024 18:51|Debit Card|Unit 4248 Box 34...|Books|4.70639588|216.4168113|
| 252922|A|4|17.34917877|8/26/2023 13:34|Cash|Unit 1535 Box 57...|Clothing|3.582091919|66.91086094|
| 172943|A|1|26.19405865|9/9/2023 0:45|Credit Card|Unit 9800 Box 87...|Electronics|17.69561007|21.55886017|
| 736766|D|9|34.9396491|12/16/2023 18:55|Debit Card|USNS Jackson FPO...|Electronics|10.66535065|280.918917|
| 276714|A|2|21.22533898|3/4/2024 5:49|Cash|Unit 4152 Box 68...|Clothing|0.305080616|42.32116916|
+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 10 rows

24/11/29 10:28:53 INFO org.sparkproject.jetty.server.AbstractConnector: Stopped Spark@1db7df6(HTTP/1.1, (http/1.1)){0.0.0.0:0}
24/11/29 10:28:53 INFO com.google.cloud.dataproc.DataprocSparkPlugin: Shutting down driver plugin. metrics={files_created=2, gcs_api_server_not_implemented_error_count=0, gcs_api_server_timeout_count=0, action http post request failures=0, op_get_list_status_result_size=1, op_open=0, gcs_api_client_unauthorized_response_count=0, action http head request failures=0, stream_read_close_operations=0, stream_read_bytes_backwards_on_seek=0, exception_count=29, gcs_api_total_request_count=47, op_create=2, gcs_api_client_bad_request_count=0, op_create_non_recursive=0, gcs_api_client_gone_response_count=0, stream_write_operations=0, stream_read_operations=0, gcs_api_client_request_timeout_count=0, op_rename=0, op_get_file_status=5, stream_read_total_bytes=0, op_glob_status=0, stream_read_exceptions=0, action http get request failures=0, op_exists=0, stream_write_bytes=168322, op_xattr_list=0, stream_write_exceptions=0, gcs_api_server_unavailable_count=0, directories_created=0, files_delete_rejected=0, op_xattr_get_named=0, op_hsync=0, stream_read_operations_incomplete=0, op_delete=3, stream_read_bytes=0, gcs_api_client_non_found_response_count=17, gcs_api_client_requested_range_not_satisfiable_count=0, op_hflush=0, op_list_status=1, op_xattr_get_named_map=0, gcs_api_client_side_error_count=6, op_get_file_checksum=0, action http delete request fa

```

Partie 3 : Analyse et Exploitation des Données

Pour envoyer ces données transformées nous créons une table externe et un dataset afin d'analyser sur BigQuery :

```
bq mkdef --autodetect --source_format=PARQUET
```

```
gs://data-lake-cleansed/transactions_cleaned.parquet > table_def.json
```

```
bq mk --external_table_definition=table_def.json twitter.transactions_cleaned
```

Table 'data-lake-project-442623:twitter.transactions_cleaned' successfully created.

```

bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623) $ bq show twitter.transactions_cleaned
Table data-lake-project-442623:twitter.transactions_cleaned

  Last modified   Schema                                     Type    Total URIs  Expiration  Labels
-----
25 Nov 18:37:59  |- CustomerID: string                     EXTERNAL  1
                  |- ProductID: string
                  |- Quantity: string
                  |- Price: float
                  |- TransactionDate: string
                  |- PaymentMethod: string
                  |- StoreLocation: string
                  |- ProductCategory: string
                  |- DiscountApplied__ : string
                  |- TotalAmount: string

```

Créer le fichier api flask

```

from flask import Flask, jsonify
from google.cloud import bigquery
import os

```

```
app = Flask(__name__)
```

```
@app.route('/api/transactions', methods=['GET'])
```

```
def get_transactions():
```

```

client = bigquery.Client()
#query = "SELECT * FROM
`data-lake-project-442623.dataset.transactions_cleaned_v2` LIMIT 10"
query = "SELECT * FROM
`data-lake-project-442623.new_dataset.transactions_cleaned_v2` LIMIT
5;"

#query = "SELECT * FROM
`data-lake-project-442623.new_dataset.logs_cleaned` LIMIT 5;"

results = client.query(query).result()
return jsonify([dict(row) for row in results])

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=int(os.environ.get("PORT", 5000)),
debug=True)

#app.run(debug=True) os

```

L'ancement de l'API:

```

bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623)$ python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://10.88.0.5:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 719-309-969
127.0.0.1 - - [28/Nov/2024 22:45:32] "GET /api/transactions HTTP/1.1" 200 -

```

Resultat / connexion a l'api :

```
bilgenur_ozdemir95@cloudshell:~ (data-lake-project-442623) $ curl http://127.0.0.1:5000/api/transactions
[
  {
    "CustomerID": "165164",
    "DiscountApplied": "15.07526967",
    "PaymentMethod": "PayPal",
    "Price": "63.30269792",
    "ProductCategory": "Books",
    "ProductID": "D",
    "Quantity": "1",
    "StoreLocation": "\"USS Camacho FPO AP 85469\"",
    "TotalAmount": "53.7596455",
    "TransactionDate": "11/22/2023 8:16"
  },
  {
    "CustomerID": "847596",
    "DiscountApplied": "2.708054072",
    "PaymentMethod": "PayPal",
    "Price": "33.06456094",
    "ProductCategory": "Books",
    "ProductID": "C",
    "Quantity": "1",
    "StoreLocation": "\"USNS Welch FPO AA 35854\"",
    "TotalAmount": "32.16915475",
    "TransactionDate": "12/16/2023 23:16"
  },
  {
    "CustomerID": "284133",
    "DiscountApplied": "6.111161299",
    "PaymentMethod": "PayPal",
    "Price": "84.79768177",
    "ProductCategory": "Books",
    "ProductID": "B",
    "Quantity": "1",
    "StoreLocation": "\"Unit 4037 Box 6496 DPO AE 68426\"",
    "TotalAmount": "79.61555866",
    "TransactionDate": "10/26/2023 7:42"
  },
  {
    "CustomerID": "953420",
    "DiscountApplied": "10.82956896",
    "PaymentMethod": "PayPal",
    "Price": "31.59375787",
    "ProductCategory": "Books",
    "ProductID": "A",
    "Quantity": "1",
    "StoreLocation": "\"USCGC Morales FPO AP 73540\"",
    "TransactionDate": "10/26/2023 7:42"
  }
]
```

L'API renvoie une requete pour l'affichage des 5 premieres.

Enfin , pour afficher des graphiques pour l'analyse et l'exploitation des données sur BigQuery, nous utilisons **Google Data Studio**, les **Graphiques intégrés dans BigQuery**,

CSV

Google Cloud

data-lake-project

Tapez / pour rechercher des ressources, des documents, des produits, etc

Recherche

État de l'essai gratuit : il vous reste 86 jours et 247,48 € de crédit. Activez votre compte complet pour bénéficier d'un accès illimité à l'ensemble de Google Cloud. Utilisez les crédits restants, puis ne payez que pour les ressources que vous utilisez.

MASQUER

ACTIVER

BigQuery

Explorateur

+ AJOUTER

Rechercher des ressources BigQuery

Afficher les ressources

NAFFICHER QUE LES FAVORIS

Préparations de données

Workflows

Connexions externes

dataset

dataset_name

new_dataset

logs_cleaned

transactions_cleaned_v2

twitter

transactions...

REQUÊTE

SCHÉMA

DÉTAILS

APERÇU

EXPLORATEUR DE TABLES

BÊTA

STATISTIQUES

Filter

Saisissez le nom ou la valeur de la propriété

Nom du champ	Type	Mode	Clé	Classement	Valeur par défaut
CustomerID	STRING	NULLABLE	-	-	-
ProductID	STRING	NULLABLE	-	-	-
Quantity	STRING	NULLABLE	-	-	-
Price	STRING	NULLABLE	-	-	-
TransactionDate	STRING	NULLABLE	-	-	-
PaymentMethod	STRING	NULLABLE	-	-	-
StoreLocation	STRING	NULLABLE	-	-	-

CLOUD SHELL

Terminal

(data-lake-project-442623)

Ouvrir l'éditeur

TransactionCount par PaymentMethod



Exemple d'analyse via les requetes :

nombre_de_transaction_par_met_de_paiement

EXÉCUTER

1 SELECT PaymentMethod, COUNT(*) as TransactionCount

2 FROM `new_dataset.transactions_cleaned_v2`

3 GROUP BY PaymentMethod

4 ORDER BY TransactionCount DESC;

Appu

Résultats de la requête

ENREGISTRER LES RÉSULTATS

	INFORMATIONS SUR LE JOB	RÉSULTATS	GRAPHIQUE	JSON
Ligne	PaymentMethod	TransactionCount		
1	Cash	1839		
2	Debit Card	1779		
3	Credit Card	1777		
4	PayPal	1733		

Graphique :

 nombre_de_transaction_par_met_de_paiement

 EXÉCUTER

```
1 SELECT PaymentMethod, COUNT(*) as TransactionCount
2 FROM `new_dataset.transactions_cleaned_v2`
3 GROUP BY PaymentMethod
4 ORDER BY TransactionCount DESC;
```


Appu

Résultats de la requête


[ENREGISTRER LES RÉSULTATS](#)

INFORMATIONS SUR LE JOB				RÉSULTATS	GRAPHIQUE	JSON
Ligne	PaymentMethod		TransactionCount			
1	Cash		1839			
2	Debit Card		1779			
3	Credit Card		1777			
4	PayPal		1733			

log

 data-lake-project

Tapez / pour rechercher des ressources, des documents, des produits, etc

 Recherche

État de l'essai gratuit : Il vous reste 86 jours et 247,48 € de crédit. Activez votre compte complet pour bénéficier d'un accès illimité à l'ensemble de Google Cloud. Utilisez les crédits restants, puis ne payez que pour les ressources que vous utilisez.

BigQuery

Explorateur

Ajouter

Rechercher des ressources BigQuery

Afficher les ressources

NAFFICHER QUE LES FAVORIS

Préparations de données

Workflows

Connexions externes

dataset

dataset_name

new_dataset

logs_cleaned

transactions_cleaned_v2

twitter

RÉSUMÉ

logs_cleaned

REQUÊTE

SCHÉMA

DÉTAILS

APERÇU

EXPLORATEUR DE TABLES BÉTA

STATISTIQUES

Filter

Saisissez le nom ou la valeur de la propriété

Nom du champ	Type	Mode	Cli	Classement	Valeur par défaut	Ta
IP	STRING	NULLABLE	-	-	-	-
Timestamp	STRING	NULLABLE	-	-	-	-
Request_Type	STRING	NULLABLE	-	-	-	-
Endpoint	STRING	NULLABLE	-	-	-	-

MODIFIER LE SCHÉMA

AFFICHER LES RÈGLES D'ACCÈS AUX LIGNES

CLOUD SHELL

Terminal

(data-lake-project-442623)

Outil Capture d'écran

Requête sans titre

EXÉCUTER

ENREGISTRER

TÉLÉCHARGER

PARTAGER

Requête terminée

1

SELECT * FROM `data-lake-project-442623.new_dataset.logs_cleaned`

Appuyez sur Alt+F1 pour afficher les options d'accessibilité

Résultats de la requête

ENREGISTRER LES RÉSULTATS

EXPLORER LES DONNÉES

INFORMATIONS SUR LE JOB

RÉSULTATS

GRAPHIQUE

JSON

DÉTAILS DE L'EXÉCUTION

GRAPHIQUE

Ligne	IP	Timestamp	Request_Type	Endpoint
1	67.214.178.190	[17/May/2015:10:05:48	"GET	/
2	123.125.71.117	[17/May/2015:11:05:16	"GET	/
3	220.181.108.153	[17/May/2015:11:05:09	"GET	/
4	65.19.138.34	[17/May/2015:11:05:40	"GET	/
5	5.255.72.168	[17/May/2015:11:05:21	"GET	/
6	82.247.137.135	[17/May/2015:12:05:29	"GET	/
7	144.76.194.187	[17/May/2015:13:05:33	"GET	/
8	166.147.88.15	[17/May/2015:15:05:33	"GET	/
9	183.60.243.242	[17/May/2015:15:05:26	"GET	/

Partie 4 : Sécurité, Gouvernance et Qualité des Données

Pour la sécurisation des données on attribue des rôles et accessibilités uniquement par les personnes autorisées :

- Configurez les rôles IAM :

```
bash
```

```
gcloud projects add-iam-policy-binding data-lake-project-442623 \
```

```
--member="serviceAccount:data-lake-project-442623-compute@developer.gserviceaccount.com" \
```

```
--role="roles/bigquery.dataViewer"
```

Ou bien vérifier manuellement

Google Cloud data-lake-project IAM Recherche

État de l'essai gratuit : il vous reste 86 jours et 246,22 € de crédit. Activez votre compte complet pour bénéficier d'un accès illimité à l'ensemble de Google Cloud. Utilisez les crédits restants, puis ne payez que pour les ressources que vous utilisez. MASQUER ACTIVER

IAM et administration IAM APPRENDRE

AUTORISER REFUSER HISTORIQUE DES RECOMMANDATIONS

ACCORDER L'ACCÈS SUPPRIMER L'ACCÈS

Filtrer Saisissez le nom ou la valeur de la propriété

Type	Compte principal	Nom	Rôle	Ins...
<input type="checkbox"/>	1040140614716-compute@developer.gserviceaccount.com	Compute Engine default service account	Lecteur de données BigQuery Propriétaire	
<input type="checkbox"/>	bilgenur.ozdemir95@gmail.com	Bilgenur Ozdemir	Lecteur de données BigQuery Propriétaire	

CLOUD SHELL

<https://console.cloud.google.com/iam-admin/iam?cloudshell=true&hl=fr&inv=A...> Ouvrir le terminal

-----Capture des graphiques -analyse -----

impact_remise_vente EXÉCUTER ENREGISTRER LA REQUÊTE

```

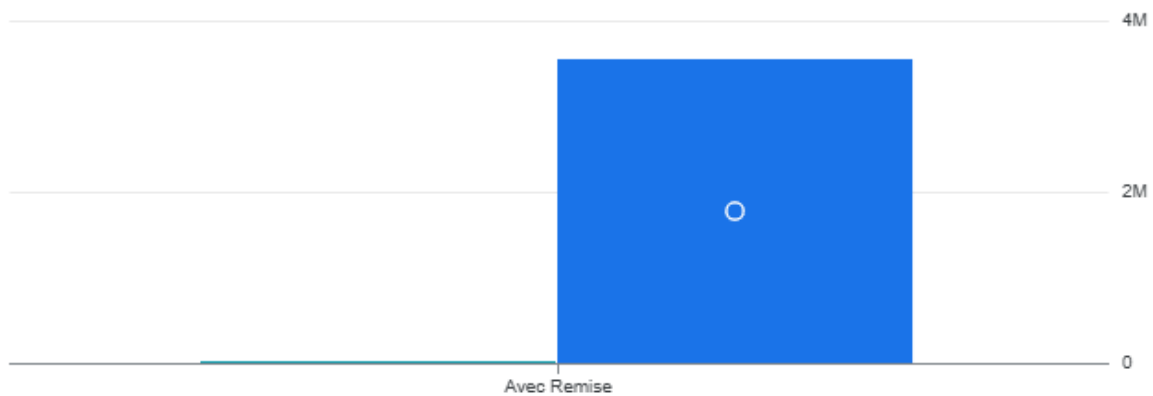
1 SELECT
2   CASE
3     WHEN SAFE_CAST(DiscountApplied AS FLOAT64) > 0 THEN 'Avec Remise'
4     ELSE 'Sans Remise'
5   END AS DiscountStatus,
6   COUNT(*) AS TransactionCount,
7   SUM(SAFE_CAST(TotalAmount AS FLOAT64)) AS TotalSales
8 FROM `new_dataset.transactions_cleaned_v2`
9 GROUP BY DiscountStatus;
10

```

Résultats de la requête ENREGISTRER LES R

INFORMATIONS SUR LE JOB		RÉSULTATS	GRAPHIQUE	JSON	DÉTAILS D
Ligne	DiscountStatus	TransactionCount	TotalSales		
1	Avec Remise	14256	3555364.148773...		

TransactionCount, TotalSales par DiscountStatus



nombre_de_transaction_par_met_de_paiement

EXÉCUTER

```
1 SELECT PaymentMethod, COUNT(*) as TransactionCount
2 FROM `new_dataset.transactions_cleaned_v2`
3 GROUP BY PaymentMethod
4 ORDER BY TransactionCount DESC;
5
```

Résultats de la requête

ENREGISTRER

INFORMATIONS SUR LE JOB

RÉSULTATS

GRAPHIQUE

JSON

D

Ligne	PaymentMethod	TransactionCount
1	Cash	3678
2	Debit Card	3558
3	Credit Card	3554
4	PayPal	3466

TransactionCount par PaymentMethod



🔍 top_10_depenser_le_+

▶ EXÉCUTER

📄 ENREGISTRER LA REQUÊTE ▼

```

1 SELECT
2   CustomerID,
3   SUM(SAFE_CAST(TotalAmount AS FLOAT64)) AS TotalSpent
4 FROM `new_dataset.transactions_cleaned_v2`
5 GROUP BY CustomerID
6 ORDER BY TotalSpent DESC
7 LIMIT 10;
8

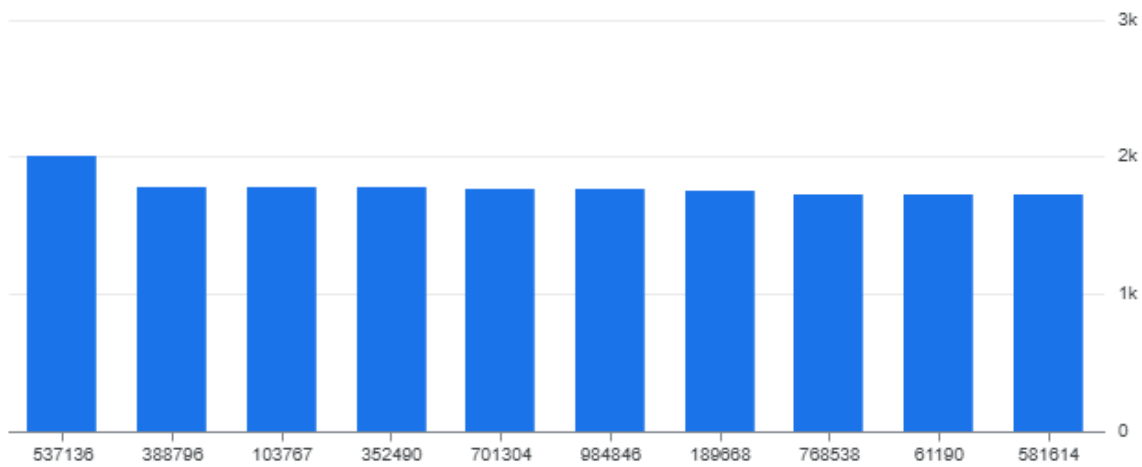
```

Résultats de la requête

📄 ENREGISTRER LES RÉSULTATS

INFORMATIONS SUR LE JOB		RÉSULTATS	GRAPHIQUE	JSON	DÉTAILS I
Ligne	CustomerID ▼	TotalSpent ▼			
1	537136	2006.828871000...			
2	388796	1784.4803072			
3	103767	1776.8351988			
4	352490	1775.773561			

TotalSpent par CustomerID



total_de_vente_par_categorie_de_produit

EXÉCUTER

```
1 SELECT
2   ProductCategory,
3   SUM(SAFE_CAST(TotalAmount AS FLOAT64)) AS TotalSales
4 FROM `new_dataset.transactions_cleaned_v2`
5 GROUP BY ProductCategory
6 ORDER BY TotalSales DESC;
7
```

Résultats de la requête

ENREGIST

INFORMATIONS SUR LE JOB

RÉSULTATS

GRAPHIQUE

JSON

Ligne	ProductCategory	TotalSales
1	Clothing	917117.3796691...
2	Books	905722.6089505...
3	Electronics	891748.3021964...
4	Home Decor	840775.8579578...

TotalSales par ProductCategory

