ETL Mainframe → AWS Data Lake → IBM (DB2/MQ)

ETL Mainframe → AWS Data Lake → IBM (DB2/MQ)
==========

Diagrama (Mermaid)

```mermaid
flowchart LR
    A[Mainframe - Arquivos VSAM] -->|Extração via JCL| B[Payload JSON]
    B --> C[AWS S3 - Data Lake]
    C --> D[Glue Jobs - Spark/PySpark]

    D --> E[Dados Transformados]
    E --> F[Athena/Analysis]
    E --> G[IBM DB2]
    E --> H[IBM MQ]
```

Objetivo

- Extrair dados de tabelas/arquivos no Mainframe (ex.: VSAM)
- Transformar em payloads JSON/Parquet adequados para processamento em cloud
- Carregar no Data Lake AWS (S3 + Glue + Athena)
- Migrar/Integrar dados consistidos para IBM DB2 (persistência) e IBM MQ (mensageria)

Arquitetura & Componentes

- Origem (Mainframe): VSAM/arquivos + rotinas JCL para exportação
- Zona de Dados (S3): landing (bronze), processed (prata), curated (ouro), particionado
- Catálogo (Glue Data Catalog): bancos e tabelas, crawlers, schemas
- ETL (AWS Glue / Spark): jobs PySpark para validação, normalização e escrita Parquet
- Consulta (Athena): validação e análises ad hoc/BI
- Destino (IBM):
  - DB2 via JDBC (persistência/consulta operacional)
  - MQ via produtores (publicação de eventos/mensagens)

Padrões de Organização (S3)
---
s3://{bucket}/
  landing/source=mainframe/system=vsam/dt_proc=YYYY-MM-DD/*.json
  processed/domain=cartoes/table=transacoes/dt_ref=YYYY-MM-DD/*.parquet
  curated/mart=risco/table=exposicao/dt_ref=YYYY-MM-DD/*.parquet

- Particionamento: dt_ref (ou year/month/day)
- Formato: Parquet (compressão Snappy) para consultas rápidas no Athena
- Nomenclatura: domain/table (dados operacionais), mart/table (dados analíticos)

Segurança & Governança

ETL Mainframe → AWS Data Lake → IBM (DB2/MQ)

- Criptografia: S3 (SSE-KMS), Glue/Athena com KMS, Secrets Manager para credenciais JDBC
- Acesso: IAM/Lake Formation (least privilege). VPC Endpoints para S3/Glue/Athena/JDBC
- Qualidade: validações de esquema, contagem de linhas, checks de nulos/chaves
- Auditoria: CloudWatch Logs/Metrics, AWS CloudTrail

Instalação (Passo a passo)
---
1) Extração (Mainframe → JSON em S3)
JCL (exemplo) para exportar VSAM → flat e transferir:

```jcl
//EXTRACAO JOB (ACCT),'VSAM->FLAT',CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
/* ... REPRO INFILE(VSAM) OUTFILE(FLAT) ... */
```

2) Transformação (Glue / PySpark)
Job PySpark — leitura JSON (landing) → normalização → Parquet (processed):

```python
import sys
from pyspark.sql import SparkSession, functions as F, types as T

spark = SparkSession.builder.getOrCreate()
df = spark.read.json("s3://meu-bucket/landing/.../")
df = df.withColumn("dt_ref", F.to_date(F.col("data_raw"), "yyyyMMdd"))
df.write.mode("overwrite").partitionBy("dt_ref").parquet("s3://meu-bucket/processed/.../")
```

3) Catálogo & Consulta (Glue Crawler + Athena)

```sql
SELECT dt_ref, count(*) AS qtd, sum(valor) AS total
FROM "etl_mainframe"."cartoes_transacoes"
GROUP BY 1
ORDER BY 1;
```

4) Carga (IBM DB2) via JDBC

```python
jdbc_url = "jdbc:db2://db2.example.corp:50000/PROD"
(df.write.format("jdbc").option("url", jdbc_url)
   .option("dbtable", "SCHEMA.EXPOSICAO")
   .option("user", "<USUARIO>").option("password", "<SENHA>")
   .mode("append").save())
```

5) Publicação (IBM MQ)

```python
# Exemplo com pymqi
queue.put(b'{"tipo":"transacao","status":"processado"}')
```

ETL Mainframe → AWS Data Lake → IBM (DB2/MQ)

```

Exemplo simples (pandas) — prototipagem local
---
```python
import pandas as pd
df = pd.read_json("input/payload.json")
df["data_processada"] = pd.to_datetime(df["data_raw"], format="%Y%m%d")
df.to_parquet("output/dados_transformados.parquet")
```

Orquestração & Monitoração

- Agendamento: AWS Glue Workflows ou Amazon Step Functions
- Monitoração: CloudWatch Logs/Metrics, alarmes (falha de job, atraso de partição)
- Qualidade: checagens automáticas (contagem, duplicidade, schema drift) + Athena
- Reprocessamento: por dt_proc/dt_ref (idempotência), isolamento por partição

Esquema & Padrões de Dados

- Tipos: normalizar datas (yyyy-MM-dd), valores numéricos (Decimal(18,2))
- Chaves: id_transacao (PK lógica), dt_ref (partição e filtro)
- Campos obrigatórios: rejeitar/sinalizar registros com id_transacao nulo ou valor < 0 quando inválido

Operação (Runbook)
---
1. Chegar arquivo na landing/ para dt_proc = HOJE
2. Executar Glue Job de transformação → escrever Parquet em processed/
3. Crawler (ou MSCK REPAIR TABLE) para atualizar partições
4. Validações no Athena (qtd/total, nulos, duplicados)
5. Carga em DB2 e publicação no MQ
6. Registrar execução (CloudWatch/Glue metrics) e abrir incidente se houver erro

Checklists

Segurança
- S3 com SSE-KMS e bucket policy restritiva
- Acesso via VPC Endpoints (S3/Glue/Athena/Secrets)
- Segredos no Secrets Manager
- Perfis IAM com least privilege

Performance
- Parquet + particionamento por dt_ref
- Repartition adequado ao volume (evitar pequenos arquivos)
- Catálogo atualizado

Confiabilidade
- Retries/backoff nos jobs
- DLQ para MQ
```

- Alarmes CloudWatch

IaC (opcional)

S3 + KMS (CLI esqueleto)
```bash
aws s3api create-bucket --bucket meu-bucket --region sa-east-1
aws kms create-key --description "KMS Data Lake"
```

Glue Crawler (CLI esqueleto)
```bash
aws glue create-crawler --name crawler-processed --role arn:aws:iam::<acc>:role/GlueRole \
  --database-name etl_mainframe --targets S3Targets=[{Path="s3://meu-bucket/processed/"}]
```

Evidências / Resultados (incluir prints)

- Execução do AWS Glue (Job Run)
- Estrutura do Data Lake no S3
- Consulta Athena
- Registros no DB2 e mensagens no MQ

(Documento gerado automaticamente em 2025-08-21.)