

Guide Complet : Data Lakehouse Local avec Dremio, MinIO et Power BI

Ce guide explique comment déployer et utiliser une architecture Data Lakehouse complète en local à l'aide de Docker.

Objectif : Mettre en place un environnement fonctionnel pour stocker des données (MinIO), les requêter avec du SQL ultra-rapide (Dremio), et les analyser avec des outils de Data Science (Jupyter) et de Business Intelligence (Power BI).

Table des Matières

- [Guide Complet : Data Lakehouse Local avec Dremio, MinIO et Power BI](#)
 - [Table des Matières](#)
 - [1. Architecture des Composants](#)
 - [2. Prérequis et Installation](#)
 - [A. Installer Docker Desktop](#)
 - [B. Récupérer le Projet](#)
 - [C. \(Optionnel\) Télécharger les Images Manuellement](#)
 - [3. Démarrage et Accès aux Services](#)
 - [A. Lancer l'Environnement](#)
 - [B. Tableau de Bord des Accès](#)
 - [4. Configuration Initiale](#)
 - [A. MinIO : Créer un Bucket et Ajouter des Données](#)
 - [B. Dremio : Connecter la Source de Données MinIO](#)
 - Onglet "General"
 - Onglet "Advanced Options"
 - [5. Analyse et Exploration des Données](#)
 - [A. Avec Dremio \(Requêtes SQL\)](#)
 - [B. Avec JupyterLab \(Python\)](#)
 - [C. Avec Power BI \(Visualisation\)](#)
 - [6. Gestion de l'Environnement Docker](#)

1. Architecture des Composants

Service	Rôle	Accès Local
MinIO	Data Lake (Stockage d'objets S3)	http://localhost:9001
Dremio	Data Lakehouse (Moteur de requête SQL)	http://localhost:9047
JupyterLab	Data Science (Analyse en Python)	http://localhost:8888

2. Préquis et Installation

A. Installer Docker Desktop

Assurez-vous que Docker Desktop est installé et en cours d'exécution.

- **Windows** : [Lien de téléchargement](#)
- **Mac** : [Lien de téléchargement](#)

Installez-le et redémarrez votre ordinateur si nécessaire.

B. Récupérer le Projet

Clonez ce dépôt Git dans un dossier de votre choix.

```
git clone https://github.com/bkablam11/demo-minio-dremio.git
```

Important : Naviguez dans le dossier du projet avant de lancer toute autre commande.

```
cd demo-minio-dremio
```

C. (Optionnel) Télécharger les Images Manuellement

Pour éviter les erreurs de réseau ([TLS handshake timeout](#)), vous pouvez télécharger les images Docker en amont.

```
docker pull dremio/dremio-oss
docker pull minio/minio
docker pull jupyter/scipy-notebook
```

3. Démarrage et Accès aux Services

A. Lancer l'Environnement

Utilisez Docker Compose pour démarrer tous les services en arrière-plan.

```
docker-compose up -d
```

Vérification : Ouvrez Docker Desktop pour voir les conteneurs [minio](#), [dremio](#), et [jupyter](#) allumés en vert.

B. Tableau de Bord des Accès

Service	URL	Utilisateur	Mot de passe
Dremio	http://localhost:9047	(à créer au premier lancement <i>bkablam11</i>)	(à créer au premier lancement <i>Mercimaman2010@</i>)
MinIO	http://localhost:9001	minioadmin	minioadmin
JupyterLab	http://localhost:8888	-	password

4. Configuration Initiale

A. MinIO : Créer un Bucket et Ajouter des Données

- Accédez à MinIO : <http://localhost:9001>.
- Connectez-vous avec *minioadmin* / *minioadmin*.
- Créez un **Bucket** (ex: *datalake*).
- Uploadez des fichiers de données (CSV, Parquet...) dans ce bucket. Vous pouvez aussi simplement glisser/déposer des fichiers dans le dossier [./minio-data/datalake](#) sur votre ordinateur.

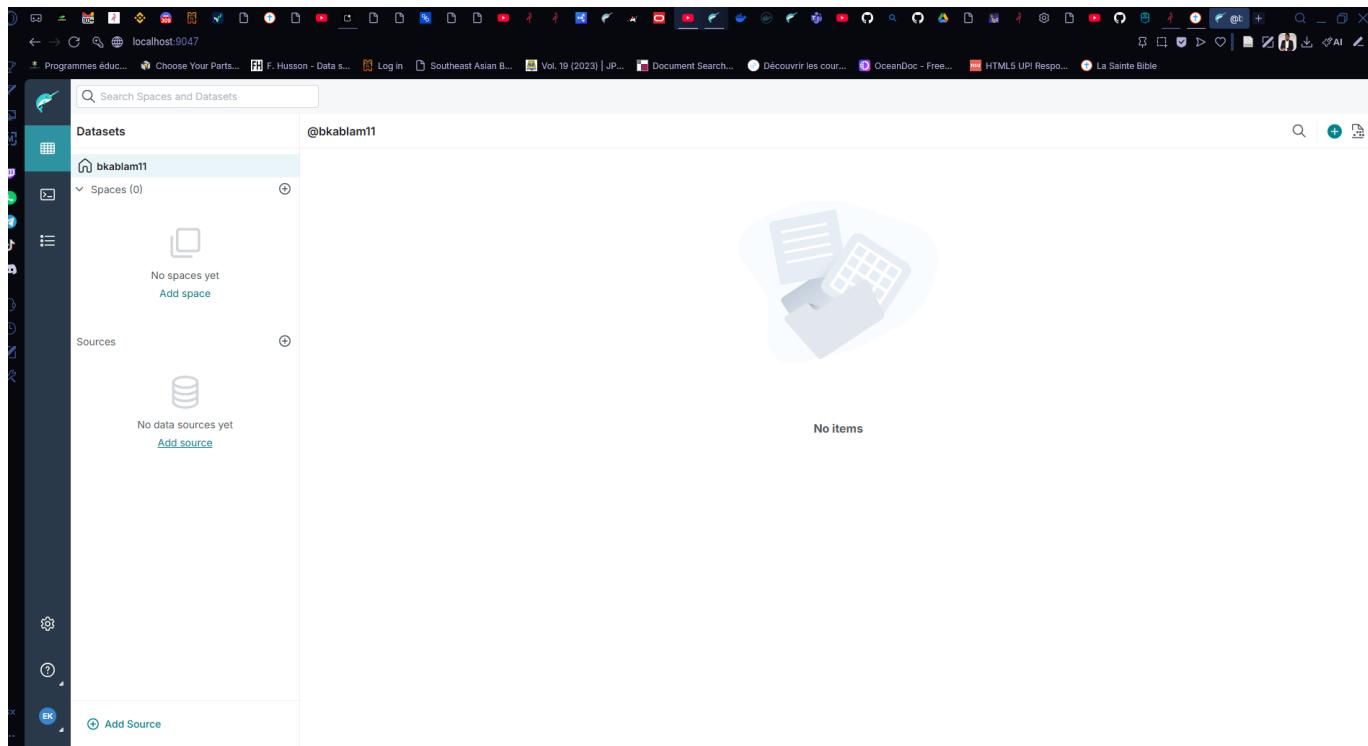
The screenshot shows the MinIO Object Browser interface. On the left, there's a sidebar with options like 'Create Bucket', 'Filter Buckets', and a list of existing buckets ('datalake', 'datalakehouse'). The main area is titled 'Object Browser' and shows a bucket named 'datalake'. Inside 'datalake', there's a single file named 'iris.csv'. The table view includes columns for 'Name', 'Last Modified', and 'Size'.

Name	Last Modified	Size
iris.csv	Today, 15:04	3.9 KIB

B. Dremio : Connecter la Source de Données MinIO

C'est l'étape cruciale où Dremio se connecte à MinIO.

- Accédez à Dremio : <http://localhost:9047> et créez votre compte administrateur.
- Cliquez sur **Add Source** et sélectionnez **Amazon S3**.



3. Remplissez les onglets comme suit :

Onglet "General"

- **Name :** MinioData (ou le nom de votre choix)
- **Authentication :** AWS Access Key
- **AWS Access Key :** minioadmin
- **AWS Access Secret :** minioadmin
- **Encrypt connection :** DÉCOCHEZ CETTE CASE (obligatoire pour une connexion HTTP locale).

New Amazon S3 Source

General

Amazon S3 Source

Advanced Options

Reflection Refresh

Metadata

Authentication

AWS Access Key EC2 Metadata AWS Profile No Authentication

All or allowlisted (if specified) buckets associated with this access key or IAM role to assume (if specified) will be available.

AWS Access Key

admin

AWS Access Secret

IAM Role to Assume

Encrypt connection

Public Buckets

Buckets

Save

Onglet "Advanced Options"

Ajoutez les propriétés de connexion suivantes pour pointer vers votre conteneur MinIO local.

Name (Nom)	Value (Valeur)	Explication
fs.s3a.endpoint	minio:9000	Adresse du conteneur MinIO dans le réseau Docker.
fs.s3a.path.style.access	true	Format d'URL compatible avec MinIO.
dremio.s3.compat	true	Mode de compatibilité S3 de Dremio.

⚠️ Attention : N'utilisez jamais `localhost:9000`. Les conteneurs communiquent via leur nom de service sur le réseau Docker.

4. Cliquez sur **Save**. Votre stockage est maintenant visible dans Dremio.

The screenshot shows the 'Source Settings' dialog in Dremio. The 'Advanced Options' tab is selected. Under 'Connection Properties', three properties are listed:

- fs.s3a.endpoint: Value minio:9000
- fs.s3a.path.style.access: Value true
- dremio.s3.compat: Value true

At the bottom right of the dialog are 'Cancel' and 'Save' buttons.

5. Analyse et Exploration des Données

A. Avec Dremio (Requêtes SQL)

1. Transformer un Fichier en Table :

- Dans Dremio, naviguez jusqu'à votre fichier CSV dans la source **MinioData**.
- Cliquez sur l'icône de formatage à droite, configurez le délimiteur (ex: `,`) et cochez **Extract Field Names**.
- Sauvegardez. L'icône du fichier devient violette, indiquant qu'il est prêt à être requêté.

2. Exécuter des Requêtes SQL :

- Ouvrez le **SQL Runner** et exécutez des commandes sur vos données.

```
-- Requête simple
SELECT * FROM MinioData.datalake."iris.csv" LIMIT 10;

-- Agrégation
SELECT species, COUNT(*) as count
FROM MinioData.datalake."iris.csv"
GROUP BY species;
```

abc A	... abc B	... abc C	... abc D	... abc E
sepal.length	sepal.width	petal.length	petal.width	variety
5.1	3.5	1.4	.2	Setosa
4.9	3	1.4	.2	Setosa
4.7	3.2	1.3	.2	Setosa
4.6	3.1	1.5	.2	Setosa
5	3.6	1.4	.2	Setosa
5.4	3.9	1.7	.4	Setosa
4.6	3.4	1.4	.3	Setosa
5	3.4	1.5	.2	Setosa
4.4	2.9	1.4	.2	Setosa

3. Créer une Vue (Virtual Dataset) :

- Après avoir exécuté une requête, cliquez sur **Save View As...** pour sauvegarder votre logique d'analyse sans dupliquer les données.

B. Avec JupyterLab (Python)

- Accès :** Allez sur <http://localhost:8888> et entrez le mot de passe **password**.
- Créer un Notebook** et utilisez le code suivant pour vous connecter à MinIO et charger des données dans un DataFrame Pandas.

```
# 1. Installation des librairies nécessaires
!pip install s3fs pandas

import pandas as pd

# 2. Configuration de la connexion à MinIO
storage_options = {
    "key": "minioadmin",
    "secret": "minioadmin",
    "client_kwargs": {
        "endpoint_url": "http://minio:9000"
    }
}
```

```
# 3. Lecture d'un fichier depuis un bucket
try:
    df = pd.read_csv('s3://datalake/iris.csv', storage_options=storage_options)
    print("✅ Données chargées avec succès !")
    display(df.head())
except Exception as e:
    print(f"❌ Erreur de connexion : {e}")
```

The screenshot shows a Jupyter Notebook environment with a single cell containing Python code. The code imports pandas, defines storage options for connecting to a MinIO Docker instance, and then reads an Iris CSV file from an S3 bucket using these options. It prints a success message and displays the first five rows of the DataFrame. The output cell shows the printed message and the resulting DataFrame.

```
File Edit View Run Kernel Tabs Settings Help
+ C
Filter files by name
Name Last Modified
test.ipynb 3 minutes ago
[3]: import pandas as pd
import os

# 2. Configuration de la connexion vers MinIO (Réseau Docker)
# Note : On utilise "minio" comme adresse car Jupyter est dans le même réseau Docker
storage_options = {
    "key": "minioadmin",      # User défini dans le docker-compose
    "secret": "minioadmin",   # Password défini dans le docker-compose
    "client_kwargs": {
        "endpoint_url": "http://minio:9000"
    }
}

# 3. Lecture d'un fichier CSV depuis le Bucket 'datalake'
# Remplacer 'iris.csv' par le nom de votre fichier
try:
    df = pd.read_csv('s3://datalake/iris.csv', storage_options=storage_options)
    print("✅ Connexion réussie ! Voici les premières lignes :")
    display(df.head())
except Exception as e:
    print(f"❌ Erreur : {e}")

[3]: ✅ Connexion réussie ! Voici les premières lignes :
      sepal.length  sepal.width  petal.length  petal.width  variety
      0            5.1          3.5          1.4          0.2     Setosa
      1            4.9          3.0          1.4          0.2     Setosa
      2            4.7          3.2          1.3          0.2     Setosa
      3            4.6          3.1          1.5          0.2     Setosa
      4            5.0          3.6          1.4          0.2     Setosa
```

C. Avec Power BI (Visualisation)

1. Installer le Driver ODBC :

- Téléchargez et installez le **Windows 64-bit ODBC Installer** depuis la [page des drivers Dremio](#).

2. Configurer la Source de Données ODBC :

- Dans la recherche Windows, ouvrez "**Administrateur de sources de données ODBC (64 bits)**".
- Sous **Sources de données système**, cliquez sur **Ajouter...** et sélectionnez **Dremio Connector**.
- Configurez comme suit :
 - Data Source Name :** **DremioDocker**
 - Host :** **localhost**
 - Port :** **31010**
 - Authentication :** **Plain**
 - User/Password :** Vos identifiants Dremio.
- Testez et sauvegardez la connexion.

3. Connecter Power BI :

- Dans Power BI, choisissez **Obtenir les données > Plus... > ODBC**.
- Sélectionnez **DremioDocker** (le nom de votre source de données) dans la liste déroulante.
- Naviguez jusqu'à vos données et commencez à créer vos tableaux de bord.

The screenshot shows the Dremio Data Explorer interface. At the top, there's a search bar with 'Enable SSO to Dremio from Power BI' and a URL bar with 'dremio.org/space/Business/Transportation."NYC%20Trips"?tipVersion=0006379312183110&version=0006379312183110'. Below the header, the 'Data' tab is selected. A sidebar on the left lists datasets like 'NYC Trips' under 'Business, Transportation'. The main area shows a SQL query: '1 SELECT * FROM "NYC Trips"'. Below the query is a preview of the data in a table format. The table has columns: vendor_id, pickup_date, pickup_datetime, dropoff_date, dropoff_datetime, passenger_count, trip_distance_mi, pickup_longitude, pickup_latitude, pickup_address, pickup_neighborhood, pickup_borough, pickup_census tract, dropoff_address, dropoff_neighborhood, dropoff_borough, dropoff_census tract, and trip_id. The data shows various trips with details like date, time, location, and distance. The preview shows approximately 60,000 records.

localhost:31010

A	B	C	D	E
sepal.length	sepal.width	petal.length	petal.width	variety
5.1	3.5	1.4	.2	Setosa
4.9	3	1.4	.2	Setosa
4.7	3.2	1.3	.2	Setosa
4.6	3.1	1.5	.2	Setosa
5	3.6	1.4	.2	Setosa
5.4	3.9	1.7	.4	Setosa
4.6	3.4	1.4	.3	Setosa
5	3.4	1.5	.2	Setosa
4.4	2.9	1.4	.2	Setosa
4.9	3.1	1.5	.1	Setosa
5.4	3.7	1.5	.2	Setosa
4.8	3.4	1.6	.2	Setosa
4.8	3	1.4	.1	Setosa
4.3	3	1.1	.1	Setosa
5.8	4	1.2	.2	Setosa
5.7	4.4	1.5	.4	Setosa
5.4	3.9	1.3	.4	Setosa
5.1	3.5	1.4	.3	Setosa
5.7	3.8	1.7	.3	Setosa

i Les données dans l'aperçu ont été tronquées en raison de limites de taille.

Charger

Transformer les données

Annuler

6. Gestion de l'Environnement Docker

- **Allumer les conteneurs actifs :**

```
docker-compose up -d
```

- **Vérifier les conteneurs actifs :**

```
docker ps
```

- **Arrêter tous les services :**

```
docker-compose down
```

| Vos données dans MinIO et votre configuration Dremio seront conservées grâce aux volumes persistants.