Source: dremio_simple_query_docs.md

dremio_simple_query

The purpose of this library is to easily query a Dremio source using Arrow Flight for analytics.

LEARN MORE ABOUT DREMIO

Use Dremio to Help:

- Govern your data
- Join your data across sources (Iceberg, Delta, S3, JSON, CSV, RDBMS, and more)
- Accelerate your queries across data sources
- Reduce your Data Warehouse Workloads

With this library your analysts can more easily get their data from Dremio and easily get to work running local analytics with Arrow, Pandas, Polars and DuckDB. This library can grab large datasets performantly thanks to using Apache Arrow Flight.

Recommended: Use the V2 Client (`dremio_simple_query.connectv2`)

The V2 client is the modern, robust implementation designed for stability and advanced features, especially for Dremio Cloud.

Why use V2?

Feature	V1 (`connect.py`)	V2 (`connectv2.py`)
Authentication	Token Only	**Token OR Username/Password** (Auto-Handshake)
Dremio Cloud	Basic Support	**Robust Support** (Session Persistence via Cookies)
Project Context	Default Project Only	**Multi-Project Support** (Route via `project_id`)
Code Quality	Basic	**Type Hinted & Docstringed**

V2 Quick Start

```
from dremio_simple_query.connectv2 import DremioConnection
from os import getenv
from dotenv import load_dotenv

load_dotenv()

# Option 1: Authenticate with PAT (Personal Access Token)
dremio = DremioConnection(
    location=getenv("ARROW_ENDPOINT"), # e.g., grpc+tls://data.dremio.cloud:443
    token=getenv("DREMIO_TOKEN"),
```

```
    project_id=getenv("DREMIO_PROJECT_ID") # Optional: Specify Project ID context
)

# Option 2: Authenticate with Username/Password (Software Only)
# Performs automatic Arrow Flight Handshake
dremio_auth = DremioConnection(
    location="grpc+tls://dremio.company.com:32010",
    username="my_user",
    password="my_password"
)

# Query Data (Returns FlightStreamReader)
stream = dremio.toArrow("SELECT * FROM star_wars.battles")

# Convert to your favorite format
df_pandas = dremio.toPandas("SELECT * FROM star_wars.battles")
df_polars = dremio.toPolars("SELECT * FROM star_wars.battles")
duck_rel  = dremio.toDuckDB("SELECT * FROM star_wars.battles")
```

Using Profiles (New in v1.0.0)

You can manage multiple environments using a `~/.dremio/profiles.yaml` file.

Setup `~/.dremio/profiles.yaml`:

```
profiles:
  my_cloud:
    type: cloud
    base_url: https://api.dremio.cloud
    project_id: 788baab4-3c3b-42da-9f1d-5cc6dc03147d
    auth:
      type: pat
      token: MY_TOKEN
  my_software:
    type: software
    base_url: https://dremio.company.com
    auth:
      type: username_password
      username: my_user
      password: my_password
default_profile: my_cloud
```

Usage:

```
# Use a specific profile
dremio = DremioConnection(profile="my_software")

# Use the default profile
dremio = DremioConnection(profile="default")
```

Manual Configuration

Legacy: Use the V1 Client (`dremio_simple_query.connect`)

The original client is maintained for backward compatibility. It is lighter but lacks session persistence features required for stable Dremio Cloud routing.

```
from dremio_simple_query.connect import DremioConnection

# Authenticate with Token Only
dremio = DremioConnection(token="MY_TOKEN", location="grpc+tls://data.dremio.cloud:443")

# Query
stream = dremio.toArrow("SELECT 1")
```

Detailed Usage Guide

Getting Your URI and Token

	Protocol	Endpoint	Result
Dremio Cloud (NA)	grpc+tls://	data.dremio.cloud:443	grpc+tls://data.dremio.cloud:443
Dremio Cloud (EU)	grpc+tls://	data.eu.dremio.cloud:443	grpc+tls://data.eu.dremio.cloud:443
Dremio Software (SSL)	grpc+tls://	`<ip-address>`:32010	grpc+tls://`<ip-address>`:32010
Dremio Software (NoSSL)	grpc://	`<ip-address>`:32010	grpc://`<ip-address>`:32010

Getting your token (V1 helper)

The `get_token` helper is available in v1 to fetch a token via REST API if you don't use the V2 handshake.

```
from dremio_simple_query.connect import get_token

login_endpoint = "http://localhost:9047/apiv2/login"
payload = {"userName": username, "password": password}
token = get_token(uri=login_endpoint, payload=payload)
```

Data Conversion Methods (Both V1 & V2)

The `.toArrow` method returns a `FlightStreamReader` object which can be converted into typical Arrow objects.

Arrow Table

```
arrow_table = stream.read_all()
```

Arrow RecordBatchReader

```
batch_reader = stream.to_reader()
```

toPandas

```
df = dremio.toPandas("SELECT * FROM arctic.table1;")
```

toPolars

```
df = dremio.toPolars("SELECT * FROM arctic.table1;")
```

Querying with DuckDB

Using the DuckDB Relation API

```
duck_rel = dremio.toDuckDB("SELECT * FROM arctic.table1")
result = duck_rel.query("table1", "SELECT * from table1").fetchall()
```

Querying Arrow Objects with DuckDB

```
import duckdb

con = duckdb.connection()
stream = dremio.toArrow("SELECT * FROM arctic.table1;")
my_table = stream.read_all()

# Zero-copy query against Arrow table
results = con.execute("SELECT * FROM my_table;").fetchall()
print(results)
```