## 📄 Technical Design Document

### Project: Mutual Fund Portfolio360 Dashboard

### Objective:

Build a data pipeline and interactive dashboard to track mutual fund investments using Snowflake, DBT, and Streamlit.

## 1. 📦 Data Sources and Raw Layer (Snowflake)

### 1.1 Tables and Structures

These tables will be created in the **RAW schema** of your Snowflake database.

#### a. investor\_master

CREATE OR REPLACE TABLE raw.investor\_master (  
 investor\_id STRING PRIMARY KEY,  
 name STRING,  
 email STRING,  
 phone STRING,  
 pan\_number STRING,  
 created\_date DATE  
);

#### Sample Insert:

INSERT INTO raw.investor\_master VALUES  
('INV001', 'Raj Mehta', 'raj@example.com', '9999988888', 'ABCDE1234F', '2023-01-01'),  
('INV002', 'Priya Kapoor', 'priya@example.com', '8888877777', 'PQRSX6789L', '2023-01-05');

#### b. mutual\_fund\_transactions

CREATE OR REPLACE TABLE raw.mutual\_fund\_transactions (  
 transaction\_id STRING PRIMARY KEY,  
 investor\_id STRING,  
 fund\_name STRING,  
 scheme\_code STRING,  
 transaction\_type STRING, -- BUY / SELL / SIP  
 amount FLOAT,  
 nav\_at\_time FLOAT,  
 units FLOAT,  
 transaction\_date DATE  
);

#### Sample Insert:

INSERT INTO raw.mutual\_fund\_transactions VALUES  
('TXN001', 'INV001', 'Axis Bluechip Fund', 'AXIS123', 'BUY', 5000, 50.0, 100, '2023-01-15'),  
('TXN002', 'INV001', 'Axis Bluechip Fund', 'AXIS123', 'BUY', 3000, 60.0, 50, '2023-03-01'),  
('TXN003', 'INV002', 'HDFC Midcap Opp', 'HDFC456', 'BUY', 4000, 40.0, 100, '2023-02-10');

#### c. nav\_history

CREATE OR REPLACE TABLE raw.nav\_history (  
 scheme\_code STRING,  
 fund\_name STRING,  
 nav\_date DATE,  
 nav\_value FLOAT  
);

#### Sample Insert:

INSERT INTO raw.nav\_history VALUES  
('AXIS123', 'Axis Bluechip Fund', '2023-01-15', 50.0),  
('AXIS123', 'Axis Bluechip Fund', '2023-03-01', 60.0),  
('AXIS123', 'Axis Bluechip Fund', '2024-06-30', 75.0),  
('HDFC456', 'HDFC Midcap Opp', '2023-02-10', 40.0),  
('HDFC456', 'HDFC Midcap Opp', '2024-06-30', 68.0);

#### d. fund\_master

CREATE OR REPLACE TABLE raw.fund\_master (  
 scheme\_code STRING PRIMARY KEY,  
 fund\_name STRING,  
 category STRING, -- Large Cap, Mid Cap, etc.  
 amc\_name STRING, -- Fund house  
 benchmark\_index STRING  
);

#### Sample Insert:

INSERT INTO raw.fund\_master VALUES  
('AXIS123', 'Axis Bluechip Fund', 'Large Cap', 'Axis AMC', 'Nifty 100'),  
('HDFC456', 'HDFC Midcap Opp', 'Mid Cap', 'HDFC AMC', 'Nifty Midcap 150');

## 2. 🧱 DBT Architecture

### 2.1 Folder Structure (DBT Core)

dbt\_mutual\_funds/  
├── models/  
│ ├── staging/  
│ │ ├── stg\_investor\_master.sql -- materialized='view'  
│ │ ├── stg\_transactions.sql -- materialized='incremental'  
│ │ ├── stg\_nav\_history.sql -- materialized='ephemeral'  
│ │ └── stg\_fund\_master.sql -- materialized='view'  
│ ├── marts/  
│ │ ├── fct\_current\_holdings.sql -- materialized='table'  
│ │ └── dim\_fund\_info.sql -- materialized='table'  
│ └── seeds/  
│ └── fund\_category\_metadata.csv -- metadata/config table  
├── snapshots/  
│ └── snap\_nav\_history.sql  
├── tests/  
│ └── custom\_tests.sql  
├── macros/  
│ └── calculate\_return\_percentage.sql  
├── docs/  
│ └── overview.md  
└── dbt\_project.yml

### 2.2 DBT Model Types

* view: lightweight and fast for intermediate staging
* table: used in marts for reporting
* incremental: loads only new data (good for transaction tables)
* ephemeral: in-memory CTE-like objects

### 2.3 Example: Incremental Model for stg\_transactions

{{  
 config(  
 materialized='incremental',  
 unique\_key='transaction\_id'  
 )  
}}  
  
SELECT \*  
FROM {{ source('raw', 'mutual\_fund\_transactions') }}  
{% if is\_incremental() %}  
 WHERE transaction\_date > (SELECT MAX(transaction\_date) FROM {{ this }})  
{% endif %}

### 2.4 Example: Ephemeral Model for stg\_nav\_history

{{ config(materialized='ephemeral') }}  
  
SELECT \* FROM {{ source('raw', 'nav\_history') }}

### 2.5 Custom Macro: calculate\_return\_percentage

-- macros/calculate\_return\_percentage.sql  
{% macro calculate\_return\_percentage(invested, value) %}  
 ROUND(({{ value }} - {{ invested }}) / NULLIF({{ invested }}, 0) \* 100, 2)  
{% endmacro %}

Used in model:

{{ calculate\_return\_percentage('total\_invested', 'current\_value') }}

### 2.6 DBT Seeds

seeds/fund\_category\_metadata.csv:

category,description  
Large Cap,High stability  
Mid Cap,Moderate risk  
Small Cap,High growth

Load with:

dbt seed

### 2.7 DBT Tests

version: 2  
  
models:  
 - name: stg\_transactions  
 columns:  
 - name: transaction\_id  
 tests:  
 - not\_null  
 - unique  
 - name: fct\_current\_holdings  
 columns:  
 - name: investor\_id  
 tests:  
 - not\_null  
 - relationships:  
 to: ref('stg\_investor\_master')  
 field: investor\_id

### 2.8 DBT Snapshots

{% snapshot snap\_nav\_history %}  
{{  
 config(  
 target\_schema='snapshots',  
 unique\_key='scheme\_code, nav\_date',  
 strategy='check',  
 check\_cols=['nav\_value']  
 )  
}}  
SELECT \* FROM {{ source('raw', 'nav\_history') }}  
{% endsnapshot %}

### 2.9 DBT Docs

In .yml:

models:  
 - name: fct\_current\_holdings  
 description: "Final table with NAV, units, returns."

Generate with:

dbt docs generate && dbt docs serve

## 3. ⚙️ Deployment + Automation

### 3.1 Snowpipe

CREATE OR REPLACE PIPE ingest\_nav\_history  
AS  
COPY INTO raw.nav\_history  
FROM @raw\_stage/nav/  
FILE\_FORMAT = (TYPE = 'CSV' FIELD\_OPTIONALLY\_ENCLOSED\_BY='"' SKIP\_HEADER=1);

### 3.2 Tasks

CREATE OR REPLACE TASK refresh\_holdings  
WAREHOUSE = compute\_wh  
SCHEDULE = 'USING CRON 0 \* \* \* \* UTC'  
AS  
 CALL run\_dbt(); -- external function or wrapper

## 4. 📊 Streamlit App

import streamlit as st  
from snowflake.snowpark.context import get\_active\_session  
  
session = get\_active\_session()  
df = session.table("fct\_current\_holdings").to\_pandas()  
  
st.title("📈 Mutual Fund Dashboard")  
st.dataframe(df)

## ✅ Summary

* Modular DBT project with views, tables, incrementals, seeds, snapshots, macros
* Automated data ingestion via Snowpipe
* Refresh via Tasks or CI/CD
* Visualize using Streamlit

Let me know if you want the GitHub structure exported or a sample profiles.yml next!

-- 👇 Using macro here

{{ calculate\_return\_percentage('SUM(amount)', 'SUM(units) \* MAX(nav\_value)') }} AS return\_percent