# **Holding Details Generator**

Developer Documentation

## **Overview**

This Python module generates holding details for Target Date Funds by combining real-world fund holdings data with synthetic portfolio-level values. The generated dataset can be used to populate the AST\_MULTIASSET\_DB.DBO.HoldingDetails table.

## **Table of Contents**

1. Dependencies
2. Setup and Configuration
3. Data Flow
4. Module Components
5. 4.1 Fetch Holdings Ticker
6. 4.2 Create Holdings Dictionary
7. 4.3 Generate Holdings Details
8. 4.4 Main Execution Function
9. 4.5 Optional Extended Table Functions
10. Key Algorithms and Design Decisions
11. Data Structure
12. Error Handling
13. Testing Considerations
14. Future Enhancements
15. Developer Checklist

## **1. Dependencies**

The module relies on the following external libraries:  
- pandas: For data manipulation and DataFrame operations  
- numpy: For numerical operations and synthetic data generation  
- yahooquery: For retrieving fund holdings and market data from Yahoo Finance  
- requests: For potential API calls to fetch data  
- tenacity: For retry logic (if extended to handle transient API errors)  
- datetime: For handling date fields

## **2. Setup and Configuration**

The script does not require complex configuration. A predefined list of fund tickers is included in the code. The random seed can be set in function parameters for reproducibility.

## **3. Data Flow**

The module follows this process:  
1. Fetch holdings for each Target Date Fund ticker.  
2. Enrich holdings with metadata and classify assets.  
3. Merge holdings with the Portfolio General Information table.  
4. Generate synthetic quantities, cost basis, and market values.  
5. Output the HoldingDetails table or an extended version for performance calculations.

## **4. Module Components**

### **4.1 Fetch Holdings Ticker**

Function: fetch\_holdings\_ticker(tickers)  
Purpose: Retrieves the underlying holdings for each fund ticker from Yahoo Finance.  
Output: DataFrame containing each fund's holdings and related attributes.

### **4.2 Create Holdings Dictionary**

Function: create\_holdings\_dictionary(holdings\_df)  
Purpose: Enhances holdings data with price, currency, quote type, and asset classification.  
Maps each fund ticker to its PRODUCTCODE.

### **4.3 Generate Holdings Details**

Function: generate\_holdings\_details(holdings\_df, num\_portfolios=10, seed=42)  
Purpose: Combines holdings with portfolio data and generates synthetic QUANTITY, COSTBASIS, and MARKETVALUE.  
Returns a clean table ready for insertion into the database.

### **4.4 Main Execution Function**

Function: main()  
Purpose: Runs the full data generation process and prints the resulting HoldingDetails table.

### **4.5 Optional Extended Table Functions**

Functions: generate\_merged\_holdings(), get\_df\_merged()  
Purpose: Produce an extended version of the HoldingDetails table with additional columns for downstream analysis.

## **5. Key Algorithms and Design Decisions**

- Data source: Yahoo Finance via yahooquery API.  
- Asset classification: Based on keyword matching in categoryName.  
- Synthetic data: Randomized quantities and discount rates for cost basis calculation.  
- Reproducibility: Random seed can be set.

## **6. Data Structure**

COLUMN TYPE DESCRIPTION  
PORTFOLIOCODE String Portfolio identifier  
TICKER String Holding ticker  
ISSUEDISPLAYNAME String Holding name  
CURRENCYCODE String Currency code  
ISSUETYPE String Security type  
PRICE Float Previous close price  
ASSETCLASSNAME String Equity / Fixed Income / Unknown  
QUANTITY Float Units held  
COSTBASIS Float Purchase cost  
MARKETVALUE Float Market value  
HISTORYDATE Date Record date

## **7. Error Handling**

Basic error handling includes default values for missing data from Yahoo Finance.  
Future enhancements could include retry logic using tenacity for transient API failures.

## **8. Testing Considerations**

- Test with different numbers of portfolios.  
- Validate asset classification logic.  
- Ensure reproducibility with a fixed seed.  
- Check that merging produces expected records.

## **9. Future Enhancements**

- Integrate direct database upload to Snowflake.  
- Improve asset classification with ML models.  
- Support for more fund providers and asset classes.  
- Add unit tests for each function.

## **10. Developer Checklist**

✓ Install required dependencies  
✓ Verify Yahoo Finance API access  
✓ Set reproducible random seed for testing  
✓ Validate data structure before database insertion  
✓ Test with multiple portfolio configurations