# PRODUCTMASTER Table Synthetic Data Generation

Overview & Purpose

The PRODUCTMASTER script generates a synthetic master table for investment products in the alternatives domain.   
It mirrors Assette’s reference schema and serves as the authoritative dimension for factsheets and joined analytics across holdings, performance, attribution, and commentary.   
Keys such as PRODUCTCODE are created once and reused as foreign keys across downstream tables. Proxy keys (PERFORMANCEACCOUNT, REPRESENTATIVEACCOUNT) support joins to performance and marketing layers.

Technical Design & Methodology

• Controlled vocabularies supplied via YAML (strategies, vehicle categories/types, share classes, single asset class = Alternatives).  
• Product codes imported from product\_codes.csv to preserve stable identifiers across the ecosystem.  
• Faker used for realistic product names and light variability; randomization is constrained to avoid nonsensical combinations.  
• Data quality rules: uniqueness of PRODUCTCODE, optional parent-child PARENTPRODUCTCODE hierarchy, and valid categorical mappings.  
• Output formatted Snowflake-ready (uppercase column names, ISO dates if present).

Execution Instructions

1) Place product\_codes.csv and product\_config.yaml (if used) in the working directory.  
2) Run the script in Colab or Python; review the preview DataFrame.  
3) Export CSV (e.g., productmaster.csv) and/or load with write\_pandas or COPY INTO to Snowflake.  
4) Store qualitative fields (fund\_description, fund\_strategy, fund\_commentary\_<Month>) if generated via GPT as part of the same record keyed by PRODUCTCODE.

Outputs & Schema

Columns: PRODUCTCODE, PRODUCTNAME, STRATEGY, VEHICLECATEGORY, VEHICLETYPE, ASSETCLASS, SHARECLASS,   
PERFORMANCEACCOUNT, REPRESENTATIVEACCOUNT, ISMARKETED, PARENTPRODUCTCODE, and optional qualitative fields   
fund\_description, fund\_strategy, fund\_commentary\_<Month>.   
The table is optimized for joins and can be merged into Snowflake to avoid duplicates on PRODUCTCODE.

Integration with Other Tables

• PRODUCTMASTER → PORTFOLIOGENERALINFORMATION via PRODUCTCODE  
• PRODUCTMASTER → HOLDINGSDETAILS via PRODUCTCODE (optional)  
• PRODUCTMASTER → narrative/factsheet tables via PRODUCTCODE  
This enables full-factsheet assembly and consistent key propagation across the warehouse.

Testing & QA Use Cases

• Validate referential integrity (no orphan PRODUCTCODE across related tables).  
• Verify categorical diversity (STRATEGY/VEHICLETYPE/SHARECLASS) and marketing flags for layout testing.  
• Confirm qualitative columns render correctly in factsheet templates (overflow, line breaks, special characters).

Next Steps & Recommendations

• Add ESG tags or labels for product-level filtering.  
• Implement lineage fields (CREATED\_AT, SOURCE, SYNTHETIC\_FLAG).  
• Provide versioned YAMLs and unit tests around key generation and parent-child hierarchies.