**User Guide for Defining Dimensional XBRL-GL Taxonomy and Structured CSV**

This guide explains how to use the available Python tools to generate a **dimensional XBRL-GL taxonomy** and to create corresponding **xBRL-CSV instance documents**. There are **two alternative approaches** depending on whether you start from a **palette taxonomy** or a **semantic model definition**.

**🔁 Two Alternative Workflows**

**🟦 Approach 1: From Existing Palette Taxonomy**

**Best for:** reverse engineering or extending an existing XBRL GL taxonomy (e.g., PWD 2016).

**✅ Step 1: Parse the Palette Taxonomy**

python xBRLGL\_ParseTaxonomy.py \

--base-dir ../XBRL-GL-PWD-2016-12-01 \

--palette case-c-b-m-u-e-t-s \

--output ../XBRL-GL-2025/XBRL\_GL\_case-c-b-m-u-e-t-s\_Structure.csv \

--lang ja \

--trace \

--debug

This creates an LHM-style CSV from the XSD definitions (tuple-based hierarchy).

**✅ Step 2: Generate Dimensional Taxonomy from Parsed LHM**

python xBRLGL\_TaxonomyGenerator.py \

../OIM-CSV/XBRL-GL-2025/LHM/XBRL\_GL\_case-c-b-m-u-e-t-s\_LHM.csv \

-b ../OIM-CSV/XBRL-GL-2025/gl-case-c-b-m-u-e-t-s \

-r AccntgEntrs \

-l ja \

-c usd \

-e utf-8-sig \

-d -v

**Outputs:**

* 📘 Palette XSD schema
* 📘 OIM schema (xBRL-CSV)
* 🧾 JSON metadata
* 🧪 Skeleton CSV file

**🟩 Approach 2: From FSM → BSM → LHM via Graph Walk**

**Best for:** building a new taxonomy from semantic definitions with specialization and modular extensions.

**✅ Step 1: Specialization — FSM to BSM**

python specialization.py \

../XBRL-GL-2025/FSM/XBRL-GL\_2025\_FSMa.csv \

../XBRL-GL-2025/BSM/XBRL-GL\_2025\_BSMa.csv \

-s ../XBRL-GL-2025/FSM/XBRL-GL\_2025\_FSM\_JPN.csv \

-l ../XBRL-GL-2025/BSM/XBRL-GL\_2025\_BSM\_JPN.csv \

-t -d

**✅ Step 2: Graph Walk — BSM to LHM**

python graphwalk.py \

../XBRL-GL-2025/BSM/XBRL-GL\_2025\_BSMa.csv \

../XBRL-GL-2025/LHM/XBRL-GL\_2025\_LHMa.csv \

-r "Accounting Entries+Accounting Entries JPN" \

-l ../XBRL-GL-2025/BSM/XBRL-GL\_2025\_BSM\_JPN.csv \

-m ../XBRL-GL-2025/LHM/XBRL-GL\_2025\_LHM\_JPN.csv \

-t -d

**✅ Step 3: Generate Dimensional Taxonomy from Graph Walk LHM**

python xBRLGL\_TaxonomyGenerator.py \

../XBRL-GL-2025/LHM/XBRL\_GL\_2025\_LHMa.csv \

-b ../OIM-CSV/XBRL-GL-2025/gl-2025 \

-r "Accntg Entrs" \

-l ja \

-c usd \

-e utf-8-sig \

-d -v

**🟨 Converting XBRL GL Instance to xBRL-CSV**

python xBRLGL\_StructuredCSV.py \

-i ../OIM-CSV/XBRL-GL-2025/ids/Customer\_Invoices.xml \

-n 2025-12-01 \

-s ../OIM-CSV/XBRL-GL-2025/LHM/xBRL\_GL\_case-c-b-m-u-e-t-s\_LHM.csv \

-o ../OIM-CSV/XBRL-GL-2025/OIM/Customer\_Invoices.csv \

-e utf-8-sig \

-d -v

**📦 Output Files Summary**

| **Output Type** | **Description** |
| --- | --- |
| .xsd | XBRL-GL dimensional schema (palette and OIM schema) |
| .json | Metadata file for xBRL-CSV |
| .csv | Skeleton CSV with header only (xBRL-CSV template) |
| structured CSV | Data extracted from XBRL GL instance into xBRL-CSV format |

**📂 Directory Overview Example**

XBRL-GL-2025/

├── FSM/

├── BSM/

├── LHM/

├── gl/

│ ├── cor/

│ └── bus/

├── OIM/

└── ids/

**🔧 Notes**

* Multiple root terms can be passed via -r "Entry A+Entry B" in graphwalk and generator scripts.
* Always verify that element, semantic\_path, and abbreviation\_path columns are correctly generated in your LHM.
* Enable -d -t flags for debug and trace when troubleshooting.