

Documentation: Saving *Cleaned Imports USA.csv* into MySQL for Semiconductor Trade Analysis

My Goal

The purpose of this step was to move the cleaned U.S. imports dataset into a relational database so that it can be queried efficiently, joined with other datasets, and analyzed at scale. By designing a precise SQL schema, I ensured that each column has the right type, that boolean values from the CSV (True/False) are preserved as text, and that trade values and quantities are stored in numeric formats suitable for aggregation.

Steps Taken

1. Creating the database

```
CREATE DATABASE semiconductors_db;
```

I created a dedicated schema called semiconductors_db to keep all semiconductor trade data organized and separate from other projects.

2. Selecting the database

```
USE semiconductors_db;
```

This ensures that all subsequent table creation and queries are executed inside the correct schema.

3. Defining the table schema

```
CREATE TABLE imports_data (  
    id BIGINT AUTO_INCREMENT PRIMARY KEY,  
    typeCode CHAR(1) NOT NULL,  
    freqCode CHAR(1) NOT NULL,  
    refPeriodId CHAR(8) NOT NULL,
```

```
refYear SMALLINT NOT NULL,  
refMonth TINYINT NULL,  
period CHAR(6) NOT NULL,  
reporterCode INT NOT NULL,  
reporterISO CHAR(3) NOT NULL,  
reporterDesc VARCHAR(150) NOT NULL,  
flowCode CHAR(1) NOT NULL,  
flowDesc VARCHAR(20) NOT NULL,  
partnerCode INT NOT NULL,  
partnerISO CHAR(3) NOT NULL,  
partnerDesc VARCHAR(150) NOT NULL,  
classificationCode VARCHAR(10) NOT NULL,  
classificationSearchCode VARCHAR(10) NULL,  
isOriginalClassification VARCHAR(5) NOT NULL DEFAULT 'True',  
cmdCode VARCHAR(20) NOT NULL,  
cmdDesc TEXT NOT NULL,  
aggrLevel SMALLINT NOT NULL,  
isLeaf VARCHAR(5) NOT NULL DEFAULT 'False',  
customsCode VARCHAR(20) NULL,  
customsDesc VARCHAR(300) NULL,  
motCode VARCHAR(10) NULL,  
motDesc VARCHAR(150) NULL,  
qtyUnitCode SMALLINT NOT NULL,  
qtyUnitAbbr VARCHAR(20) NOT NULL,  
qty DECIMAL(20,4) NULL,  
isQtyEstimated VARCHAR(5) NOT NULL DEFAULT 'False',
```

```
    altQtyUnitCode SMALLINT NULL,  
    altQtyUnitAbbr VARCHAR(20) NULL,  
    altQty DECIMAL(20,4) NULL,  
    isAltQtyEstimated VARCHAR(5) NOT NULL DEFAULT 'False',  
    netWgt DECIMAL(20,4) NULL,  
    isNetWgtEstimated VARCHAR(5) NOT NULL DEFAULT 'False',  
    isGrossWgtEstimated VARCHAR(5) NOT NULL DEFAULT 'False',  
    cifvalue DECIMAL(20,4) NULL,  
    fobvalue DECIMAL(20,4) NULL,  
    legacyEstimationFlag VARCHAR(10) NULL,  
    isReported VARCHAR(5) NOT NULL DEFAULT 'True',  
    isAggregate VARCHAR(5) NOT NULL DEFAULT 'False',  
    qty_standard DECIMAL(20,4) NULL,  
    qty_kg DECIMAL(20,4) NULL,  
    data_quality VARCHAR(20) NOT NULL  
);
```

4. Key design choices

- **Boolean fields as text:** All flags (isOriginalClassification, isLeaf, isQtyEstimated, etc.) were defined as VARCHAR(5) with defaults 'True' or 'False'. This matches the CSV values directly and avoids conversion errors.
- **Numeric precision:** Trade values (cifvalue, fobvalue) and quantities (qty, netWgt, qty_standard, qty_kg) were stored as DECIMAL(20,4). This ensures accuracy for large monetary values and fractional weights.
- **Identifiers:** Country codes (reporterCode, partnerCode) were set as INT to handle large numeric IDs (e.g., 108408577). ISO codes were stored as CHAR(3) for consistency.

- **Text fields:** Commodity descriptions (cmdDesc) were stored as TEXT to accommodate long strings, while shorter descriptors (reporterDesc, motDesc) use VARCHAR.

5. Importing the data

The cleaned CSV (*Cleaned Imports USA.csv*) was imported into the imports_data table using MySQL Workbench's import wizard. With the schema aligned to the CSV, the import succeeded without type errors.

6. Verification

A simple query validated the import:

```
SELECT COUNT(*) FROM imports_data;
```

This confirmed that 373 rows were loaded. Spot checks with `SELECT * FROM imports_data LIMIT 10;` showed correct alignment of values across columns.

Key Decisions and Rationale

- **Boolean simplification:** Storing flags as text avoids preprocessing and keeps the dataset consistent with the original CSV.
- **Precision in trade values:** Using DECIMAL ensures that financial and weight calculations remain accurate.
- **Schema transparency:** Each column name and type mirrors the CSV, making the table auditable and easy to cross-reference.
- **Future scalability:** The schema allows additional datasets (e.g., exports, other partner countries) to be imported into the same database for comparative analysis.

Data Source

[UN Comtrade](#) – *Cleaned Imports USA.csv*