Documentation: Cleaning USA TradeData Imports Dataset

My Goal

The purpose of this cleaning step was to prepare the export dataset so that it focuses on the variables that actually matter for analysis of U.S. semiconductor trade. I wanted to normalize quantities, handle missing values consistently, convert critical fields to numeric types, and produce a file that is transparent, auditable, and ready for budgetary or trade-partner analysis.

Steps Taken

1. Reading the raw dataset

```
df = pd.read_csv("TradeData_1_21_2026_21_51_10.csv")
```

I loaded the original export dataset into a pandas DataFrame. This file contained multiple fields, some of which were redundant, inconsistent, or filled with placeholders like N/A.

2. Converting key columns to numeric

```
for col in ["qty","altQty","netWgt","grossWgt","cifvalue","fobvalue","primaryValue"]:
    df[col] = pd.to_numeric(df[col], errors="coerce")
```

I explicitly converted important fields (quantities, weights, and values) into numeric types. This ensures that calculations and aggregations won't break due to string formatting issues. Using errors="coerce" safely turned invalid entries into NaN, which is easier to handle than inconsistent text.

3. Normalizing quantities

```
def normalize_qty(row):
    unit = row["altQtyUnitAbbr"] if row["altQtyUnitAbbr"] != "N/A" else
row["qtyUnitAbbr"]
```

```
    qty = row["qty"]
    if unit == "u":
        return qty
    elif unit == "1000u":
        return qty * 1000
    elif unit == "kg":
        return qty
    else:
        return np.nan

df["qty_standard"] = df.apply(normalize_qty, axis=1)
```

I created a new column qty_standard to normalize quantities across different units (u, 1000u, kg). This step ensures comparability across records.

I also added qty_kg as a direct reference to netWgt, replacing missing values with zero for consistency.

4. Converting booleans and labeling data quality

```
df["isNetWgtEstimated"] = df["isNetWgtEstimated"].astype(str).str.upper().map({"TRUE": True, "FALSE": False})

df["data_quality"] = df["isNetWgtEstimated"].map({True:"Estimated", False:"Reported"})
```

I standardized boolean fields and created a data_quality column to clearly indicate whether weight data was estimated or reported.

5. Selective cleaning

```
critical_cols = ["reporterISO","motDesc","qty","cifvalue"]

df = df.dropna(subset=critical_cols, how="all")
```

I dropped rows where all critical fields were missing, ensuring that only meaningful records remain.

For numeric fields, I filled missing values with zero. For text fields, I replaced N/A and nulls with "Unknown" to avoid ambiguity.

6. Removing duplicates

```
df = df.drop_duplicates()
```

I eliminated duplicate rows to ensure the dataset is clean and unique.

7. Saving the cleaned dataset

```
df.to_csv("Cleaned Imports USA.csv", index=False)
```

Finally, I exported the cleaned dataset. The result is a structured file that highlights the essentials: reporter, partner, transport mode, normalized quantity, and trade values.

Key Decisions and Rationale

- Weights handled carefully: I kept netWgt but replaced missing values with zero, while documenting whether data was estimated or reported.
- Quantity normalization: I created qty_standard to unify different reporting units, ensuring comparability.
- Text clarity: I replaced N/A with "Unknown" to make the dataset more human-readable.
- Selective cleaning: I dropped rows missing all critical fields, but preserved imperfect records with partial information for transparency.
- Redundancy reduced: I avoided duplicating values unnecessarily, focusing instead on normalized and interpretable fields.

Data Source: UN Comtrade