Module 3 Workbook: Step-by-Step Data Cleaning Pipeline

Operationalizing the UCI Online Retail Dataset Cleansing Strategy

Participant Name: _____ **Date:** _____

Section 1: The Core 4-Step Cleaning Architecture

Cleaning messy data without stress requires a systematic sequence. Below is the exact production-grade pipeline used to clean and transform raw datasets into stable elements ready for enterprise visualization.

Step 1: Ingestion & Structural Scanning (Error Detection)

Stream the live dataset directly into your environment and get immediate metrics on missing entries and duplicate instances.

```
df_online = pd.read_csv("https://raw.githubusercontent.com/mvandorpe/uci-online-retail/master/online_retail.csv")
```

`df_online.isnull().sum()` | `df_online.duplicated().sum()`

Step 2: Dropping Critical Null Values & Anomalies (Data Correction)

Purge records missing absolute primary identification keys (like `CustomerID`) or containing critical data errors where recovery is impossible.

```
df_step2 = df_online.dropna(subset=['CustomerID'])
```

Step 3: Logical Value Filtering (Business Rules Alignment)

Apply conditional filters to enforce validity constraints, such as stripping out transaction cancellations hidden as negative numbers.

```
df_step3 = df_step2[df_step2['Quantity'] > 0]
```

Step 4: Format Alignment & Text Normalization (Standardization)

Reconcile messy text variations, strip trailing whitespace bugs, and switch string date representations into official temporal timestamp formats.

```
df_step4 = df_step3.copy()
df_step4['Description'] = df_step4['Description'].str.strip()
df_step4['InvoiceDate'] = pd.to_datetime(df_step4['InvoiceDate'])
```

Section 2: Pipeline Execution Lab & Verification Queries

Implement the 4-step framework in your coding workspace and use your live terminal outputs to fulfill the structural engineering validations below.

Exercise 1: Documenting Initial Integrity Scan (Step 1)

Execute the Step 1 scanning commands on the live stream. Record the exact count of missing entries detected in the `CustomerID` column and total duplicate rows present:

Missing Customer IDs: _____ | Total Redundant Rows: _____

Exercise 2: Evaluating the Elimination Matrix (Step 2 & Step 3)

Calculate the row structural impact of filtering. How many total rows are discarded when you execute Step 2 and Step 3 combined? Show or explain your evaluation logic:

Exercise 3: Tracking Temporal Format Changes (Step 4)

Before running `pd.to_datetime()`, look at your column type via `df.dtypes`. What was the original data format classification of `InvoiceDate` vs its clean corrected datatype classification after running Step 4?

Exercise 4: Scripting Composite Custom Pipelines

Combine the separate steps above into one singular, highly-efficient Python function called `clean_retail_data(raw_df)` that returns a fully cleaned DataFrame in one step: